POCHE COMPUTERS

POCHE COMPUTERS

Le magazine des passionnés
de micros de poche
et de calculatrices

N°3

CHOISIR UN POCKET ?
PSION SERIES 3/3a - SHARP PC 3000
HP95LX -HP100LX - PORTFOLIO

Initiation à l'assembleur et à l'external sur HP 48

QUOI DE NEUF AU
CEBIT 94 DE HANNOVRE

LE POCKET LE PLUS
PUISSANT AU MONDE !
Le plus petit 486 existant

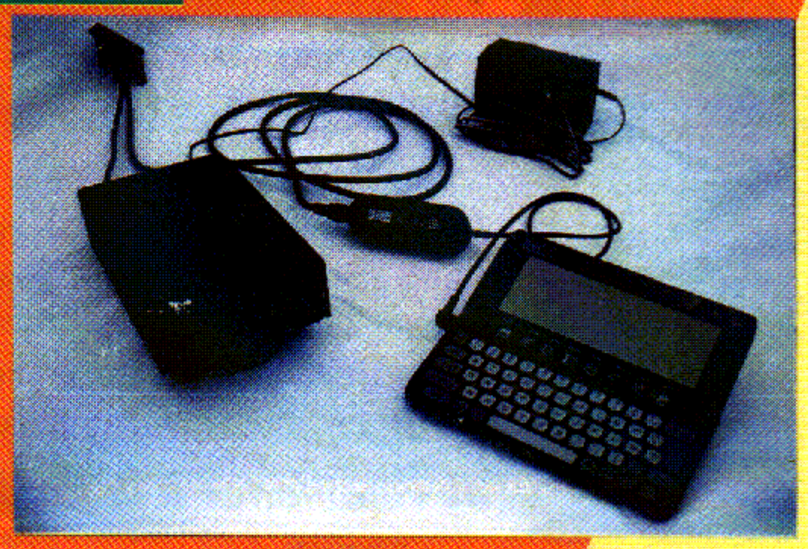
Des trucs et astuces
comme s'il en pleuvait !
pour calculatrices TI 85-CASIO
SHARP-PORTFOLIO-PSION-HP

HP100LX 2Mégas !
Communiquez avec
votre HP95LX-100LX
6 mégas de programmes
pour HP95-100 !!!

Des programmes
pour vos Casio
basic et graphiques

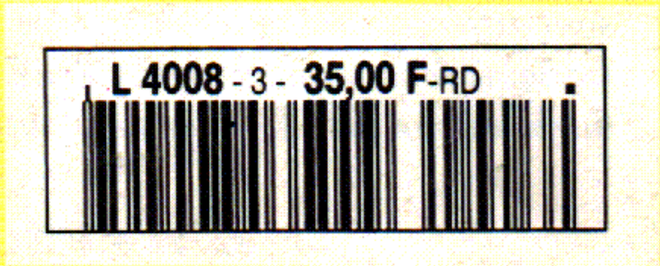
PSION series 3/3a :
essai : le lecteur de disquettes

Robotique
sur PSION



Dom-tom 39F
Belgique 256 FB
Canada 9,50 $ can
Luxembourg 245 Lux
Suisse 10,8 Frs
Italie 8500 L
Espagne 1000 pta
USA 7,5 $
Portugal 1200 esc

35F abonnement 6 numéros : 180F



Cartes mémoire RAM et ROM pour HP48 SX-GX (de 128KB à 1MB)

(compatibles 100% HP)

GARANTIE TOTALE DURANT 1 AN

Déjà plus de 2000 cartes commandées !

Cartes mémoire RAM

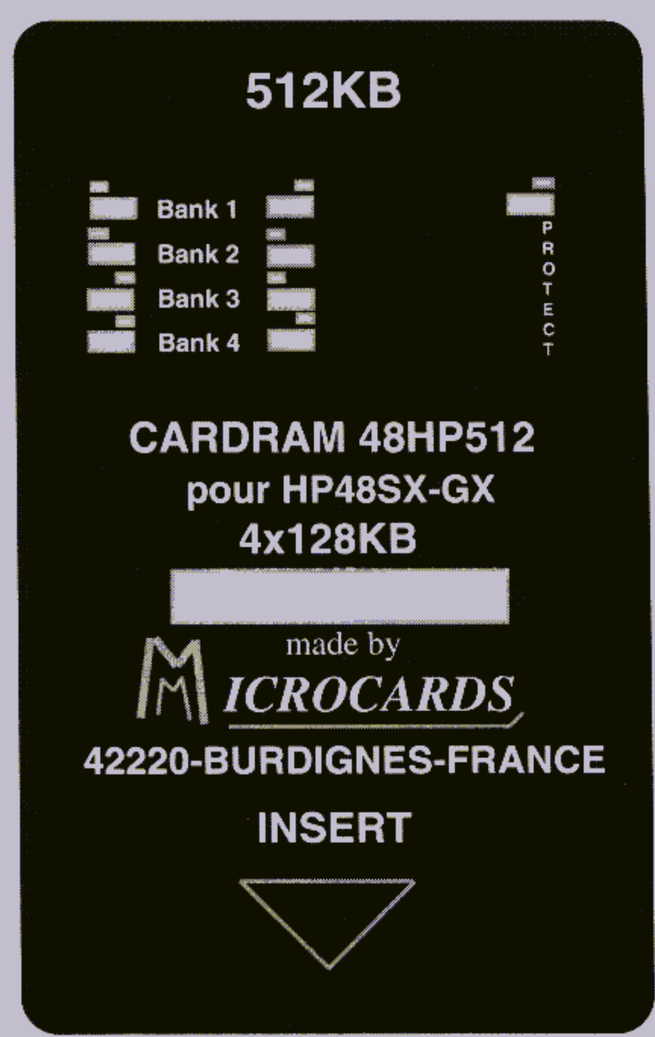
Produites par la société MICROCARDS, spécialiste en micro-électronique, en toute sécurité pour l'utilisateur.
On peut les placer indifféremment sur port 1 ou 2, et les utiliser en mémoire fusionnée ou comme mémoire annexe.
Avec nos cartes, le passage de banque est manuel, par interrupteur (ce qui permet une utilisation sur GX ou SX) ; la taille de chaque banque est de 128K octects.
Les données sont conservées par micro-accumulateur interne et non par pile, rechargeable depuis la HP, avec une sur-consommation négligeable et une durée de vie de 10 ans.
Un micro-interrupteur permet l'accès ou le blocage d'écriture.

NOUVEAUTE !!!
Chaque carte est fournie avec 2 logiciels gratuits : un logiciel de jeu et/ou utilitaire déjà chargés.

LES PRIX ? (directement du producteur au consommateur):

Cardram 48HP128	(capacité 128KB)	495F ttc
Cardram 48HP256	(capacité 256KB)	795F ttc
Cardram 48HP512	(capacité 512KB)	1295F ttc
Cardram 48HP1024	(capacité 1024KB)	2195F ttc

Frais de port colissimo recommandé: jusqu'à 1500F de commande : 45F ttc
au delà de 1500F : 60F ttc



ATTENTION : pour votre commande, précisez si vous possédez une 48SX ou une 48GX, car les logiciels sont différents selon les machines.
Certains fonctionnent sur les deux, d'autres ne fonctionnent que sur l'une ou sur l'autre.

NOUVEAU !!!

Toute une série de cartes ROM 128KB (ineffaçables), sont à présent disponibles. à des prix très bas pour le plus grand plaisir de chacun.
Vous trouverez donc pour 48SX ou 48GX:

UTILS 1: 1 carte rom utilitaires de très haut niveau !!! (Assembleur - Désassembleur - Word pour HP - Exploreur de librairies - Convertisseur de codes G/GX vers S/SX et inversement - lecture de mémoire ram et rom - Pour programmer en external - Communiquer avec le minitel etc... etc...)

JEUX 1: 1 carte rom jeux ; le must !!! (Xennon - Tetris - Boulder - Arkanoïd - Vader - Lemmings - Diamant - Columns - Solitaire - Frogg - Ice - Mazzogs - Gargage - Tunnel - etc...)

HPMANIAC 1: 1 carte rom super géniale !!! (Démineur - Editeur de tableaux - Jeu de réflexion - Reversi - DIGITAL pour écouter des sons digitalisés - MUSEDTR pour composer des musiques - PIANO pour composer sur votre clavier - BIGBANNR pour faire défiler en gros caractères - PROTECT pour protéger vos librairies - RECOVERS pour retrouver vos données perdues - PORTS pour voir le contenu des ports - MEGAZAPP une super télécommande - EXTERNAL compileur etc... etc...)

420F ttc	la carte
790F ttc	les 2 cartes
1100F ttc	les 3 cartes

Frais de port colissimo 45F ttc
(nous pouvons également vous fournir des cartes ROM 128KB avec vos propres programmes installés : prix unitaire 525F ttc
Téléphoner pour renseignements techniques)

Pour commander, utilisez de préférence notre bon de commande à la fin de ce magazine.

Copyright 1994 MICROCARDS : document non contractuel - les prix peuvent modifiés à tout moment sans avis préalable

POCHE COMPUTERS

SOMMAIRE du N°3

Juin 1994

POCHE COMPUTERS
adresse postale :
42220 BURDIGNES-FRANCE
Tél 77.39.16.33 Fax 77.39.19.60
Rédaction : P. MARTIN
Ont participé à ce numéro :
David WINTER
Etienne DEMARCQ
Thierry BODIN
Mickaël SAILLOT
Karim ALAFIA
Denis LAURENS
Philippe GOURVES
Olivier Le MÔ
Daniel GENTES
Rodolphe MENARD
Olivier ARTRES
V.LUGNIER
Karl GALOPIN
M.BARRY
Stéphane TORRES
Antony NEVEU
Maurice TALAZAC
Patrick DOUGUET
Jérôme FORESTIER
Gilles JEAN
Stéphane OUVETTE
JM POULOT
Emmanuel MEUNIER
Régis DUCHESNE
Loïc FIEUX
Anthony TARDIVEL
Noël BONNET
G.C.
JL LECOMTE
Les illustrations sont de :
Jluc MORCEAU
Publicité : au journal
Graphisme et mise en pages :
au journal
Imprimerie DUMAS
42000 ST ETIENNE
Directeur de la publication :
JL LECOMTE
Commission paritaire N° 74638
Dépôt légal : 2ème trimestre 1994
No ISSN : 1165-6905
Editeur :
MICROCARDS s.a.r.l. 63520 St Dier
Copyright 1994 MICROCARDS
Toute reproduction ou représentation intégrale ou partielle, par quelque procédé que ce soit, des pages publiées dans ce magazine sans l'autorisation écrite de l'éditeur est illicite et constitue une contrefaçon (Loi du 11 mars 1957 art. 40 et 41 et Code pénal art. 425).
Tous droits réservés pour tous pays.
Les documents reçus ne sont pas rendus et leur envoi implique l'accord tacite de l'auteur pour leur libre publication.
Les indications de marques et les adresses qui figurent dans les pages rédactionnelles de ce numéro sont données à titre d'information, sans but publicitaire.
Tous les noms de marque cités dans ce numéro sont déposés.

EDITORIAL....2
LES NEWS....3
CHOISIR UN POCKET....5
RUBRIQUE PORTFOLIO
- Jeu de cases....9
- Pdraw....10
- Test mémoire....13
- Affichage graphique....17

RUBRIQUE HP48-28
- Initiation à l'external....20
- Initiation à l'assembleur....24
- Liaison HP-Minitel....28
- Programmes....29

RUBRIQUE CASIO FX850-880P....34
TRUCS ET ASTUCES....37
RUBRIQUE SHARP....39
RUBRIQUE HP95-100LX
- Communiquez !....41
- Programmes....42

RUBRIQUE PSION
- Jeu : AWALE....43
- Essai : le lecteur de disquettes....46
- Interface de commande 8 sorties....47
- Programmes....49

RUBRIQUE TI-85
- Programmes....51

RUBRIQUE CASIO PB1000
- Mise au point photo....55

INITIATION A L'ASSEMBLEUR 8086....56
GESTIONNAIRE DE COMMANDES....61
LES LIVRES....64
RUBRIQUE CASIO GRAPHIQUES
- Animez vos graphiques Casio....65
- Programmes....67

PETITES ANNONCES....68
ABONNEMENTS....70

PROCHAINE PARUTION : Octobre 1994

EDITORIAL

Chers lecteurs et amis programmeurs, Merci. Vous êtes de plus en plus nombreux dans toute la FRANCE et maintenant dans les pays francophones à apprécier notre revue. Nous avons eu même une demande de la bibliothèque scientifique de MOSCOU ! Nous en sommes très fiers !!! Cette fois, nous avons beaucoup de retard pour la parution ; nous essaierons d'être plus ponctuels pour le numéro d'octobre prochain.
Revenant de Hannovre en Allemagne, au CEBIT, nous avons pu nous faire une idée des tendances ultérieures concernant le marché de l'informatique qui nous intéresse. Dans ce salon grand comme plusieurs fois ceux de PARIS, j'ai pu constater l' invasion (pacifique !) des asiatiques. La COREE, TAÎWAN, et la MALAISIE figurent au premier plan. Il faut dire que la main d'oeuvre est plutôt bon marché là-bas ! Le format PCMCIA est en pleine explosion, et on peut prédire aisément qu'il fera son apparition en force en FRANCE très prochainement où il est actuellement encore peu développé. Vous remarquerez également dans les "News" ces étonnantes cartes laser, dont la taille et l'épaisseur sont exactement identiques aux cartes de crédit et avec des capacités de stockage extraordinaires. Autre nouveauté, le HP 100LX version 2MB. Son prix est un peu près semblable au 1MB d'auparavant ; ce dernier baisse de 1000F environ. Hewlett Packard propose aux US une mise à niveau de votre ancien appareil pour environ 2000F. Microcards va le proposer aussi. En ce qui concerne le magazine, vous pourrez remarquer que celui-ci s'est considérablement étoffé. C'est un très long travail que de réunir toutes ces informations, de les corriger éventuellement et d'en faire la mise en page. Mais le résultat, je l'espère, est payant (concernant la qualité). Nous avons essayé de traiter d'un maximum de machines, et nous ferons encore mieux la prochaine fois ! Une dernière chose concernant de nombreux lecteurs qui se demandent comment procéder pour proposer leur articles ou idées. C'est très simple, il suffit de nous les faire parvenir par courrier! si possible tapés à la machine, ou encore mieux sous traitement de textes type PC. Nous offrons un numéro gratuit pour les petits articles et un abonnement gratuit pour les plus longs articles (à partir d'une page dactylo). Certains nous demandent si cela est rémunéré. Ceci est impossible. Je précise que POCHE COMPUTERS équilibre tout juste ses frais d'impression. Il ne perd ni ne gagne d'argent ! Le rédactionnel et la mise en page sont faits bénévolement !

CORRECTIFS DU NUMERO 2 DE POCHE COMPUTERS

La perfection n'est pas de ce monde, bien entendu. Cependant, nous nous excusons auprès de nos lecteurs de les avoir fait attendre. Nous apportons ici quelques corrections concernant 2 articles du précédent numéro. Nous en profitons une fois encore pour engager nos rédacteurs occasionnels à nous fournir leurs articles tapés à la machine ou mieux, sur disquette PC sous n'importe quel format. Cela nous évitera de très longues heures de saisie et des erreurs potentielles. La lecture de certains articles reçus tient plus du déchiffrage de hiéroglyphes que de programmes d'informatiques. Or nous ne sommes pas égyptologues ! mais informaticiens.

1) Pour le jeu de Tétris sur HP28S page 26, vous avez été nombreux à nous signaler les erreurs (de frappe) ; les voici :

Bloc numéro 1 : ...après 20202 02022 **383F2** et non 38F2
Bloc numéro 3 : ...après 2073D 2010E **42331** et non 42231
Bloc numéro 11 : ...après 70E72 15D7A **0D5DE** et non 05DE

2) Pour l'article "ANIMEZ VOS GRAPHIQUES CASIO" page 16 :

Concernant l'accès à la mémoire vive valable pour les graphiques séries 8000 et suivantes, il fallait lire :
? => EXE MODE EXP AC RANGE RANGE appuyer 7 fois sur ⇓ puis <= et DEL Eteindre la machine et rallumer, puis MODE 3 SHIFT DEL
Dans la zone de l'écran texte, remplacer n+16 par n+15
Dans la partie concernant le chiffre, la mantisse et l'exposant, page 17 colonne de gauche, il faut lire chiffre 1, mantisse +, et exposant + (et non -).
Enfin, page 17, colonne de droite dans le petit programme pour illustrer, il faut lire aux 2 premières lignes : E-12 (exposant) et non E-12.

LES NEWS

par JL LECOMTE

Voici quelques infos de notre envoyé spécial au Cebit de Hannovre 1994

Jusqu'à 6,6MB de capacité !

(l'équivalent de 3000 pages de textes !)

Après les cartes PCMCIA, voici les cartes LASER au format carte de crédit et d'une épaisseur de 0,8mm (comme une carte bancaire). Jusqu'à 50000 écritures sont possibles. On peut stocker toutes sortes de données à partir d'un PC ou d'un MAC. Cette nouvelle carte intéresse plus particulièrement le domaine bancaire, mais aussi le domaine scientifique (en particulier la médecine), le domaine commercial et bien sûr le domaine militaire. Outre son très faible encombrement, cette carte peut être produite en très grande série et à faible coût. Sa technologie s'apparente beaucoup à celle des disques laser, et d'ailleurs son aspect extérieur ressemble à un mini CD ROM, mais de forme rectangulaire. L'écriture de cette carte s'effectue à partir d'un lecteur spécial, branché sur la carte mère d'un micro, ou sur sa sortie parallèle. La société qui fabrique ce produit (USA - Californie), espère vendre 25 millions de cartes en 1994.

260MB sur une carte PCMCIA type III !

Tel est le nouveau record d'intégration et du rapport miniaturisation / puissance. Mais jusqu'où vont-ils-aller? Heureusement pour nous, utilisateurs, très loin !!! L'annonce de la société Californienne MINISTOR a fait l'effet d'une bombe (heureusement sans tuer personne !)
Ce mini disque dur est compatible avec tous les micros acceptant le type III (Acer - AST - Chaplet - Chicony - Dell -HP- Olivetti - Toshiba - Zeos - etc...)

On devait s'y attendre, eh bien c'est chose faite:

un pocket compatible PC486, le plus petit au monde vient de sortir...

Celui-ci nous vient d'extrême-orient. Pour des dimensions de 24cm de large pour 13cm de profondeur et 3,8cm d'épaisseur, il pèse 900 grammes. Son alimentation par 6 piles alcalines (ou accus rechargeables) lui donnent une autonomie de 10heures. Son affichage est un écran VGA à 64 niveaux de gris. Son processeur de type 486 SLC EV/25 permet d'accepter une mémoire centrale de 4MB jusqu'à 16MB avec un disque dur interne de 260MB. Il possède trois ports PCMCIA, deux de type 2 et un de type 3 ; track-ball intégré, interface parallèle et série, et enregistreur vocal. Sa rom contient MS-DOS 6.2 et Windows 3.1. Il est importé par MICROCARDS, existe en plusieurs versions et son prix de base est de 9900FHT.

NOUVEAU ! Carte PCMCIA type 2 SCSI

New Media vient d'annoncer la sortie d'une carte PCMCIA type1 permettant l'accès à tous les périphériques SCSI. Il est désormais possible par exemple de connecter un CD ROM à votre HP100LX !!!, mais également un scanner, un disque dur ultra rapide etc...
L'intérêt de l'interface SCSI réside par sa puissance et sa rapidité. Disponible chez MICROCARDS

LES NEWS ... LES NEWS ... LES NEWS...

DU SON POUR VOTRE POCKET...

NEW MEDIA sort également une carte PCMCIA type1, WAV jammer qui vous offre un son haute fidèlité, de qualité CD. Vous pouvez enregistrer et écouter à des cadences d'échantillonage allant jusqu'à 44,1 KHz, en mode stéréo 16 bits. La carte son est livrée avec le gestionnaire client PCMCIA de NEW MEDIA et Windows VXD, ainsi qu'avec Microsoft Sound System 2.0. Vous pouvez l'utiliser pour l'enregistrement et l'écoute à partir de la plupart des applications Windows. Elle est compatible avec AdLib et Sound Blaster.
Disponible chez MICROCARDS.

Un aperçu du stand cartes mémoire MITSUBISHI au CeBIT 94 de HANNOVRE (Allemagne).

Pour la majorité, ce sont des cartes ROM - RAM ou FLASH EPROM au standard PCMCIA type 1 ou 2 (68 pins) ; le reste étant constitué de cartes à standards spécifiques de type industriel avec des brochages de 32, 38, ou 88 pins, souvent utilisées dans des applications de robotique ou interface de communication.

Le nouveau logiciel **MONEY** pour PSION Series 3a est enfin disponible en FRANCE.
Conçu pour une utilisation de comptabilité personnelle, il propose toutes sortes de possibilités telles que :

- Gestion de compte bancaire
- Dépenses - recettes personnelles
- Livre des achats - recettes
- TVA
- Devises

etc...

Voici un jeu impressionnant de cartes mémoire PCMCIA en Flash Eprom. La société américaine SUNDISK Corporation fabrique des cartes de capacité diverses depuis 1,8MB jusqu'à 40MB, destinées aux Pockets et Portables. L'avantage des cartes Flash par rapport aux cartes SRAM est :

- une meilleure résistance aux chocs -
- une plus grande capacité en taille mémoire
- une fiabilité égale à 100%
- une conservation des données sans pile ni accumulateur
- un prix de vente inférieur

L'inconvénient majeur par rapport aux SRAM est la non possibilité de reprogrammation comme on désire. En effet, dans le cas d'une saturation mémoire, l'utilisateur doit effacer une partie de la carte pour introduire un nouveau fichier. Leur utilisation est surtout appréciable lorsqu'il s'agit de sauvegarder des fichiers très importants, tant en qualité qu'en quantité. (dispo chez MICROCARDS)

Il dépasse les limites de compression de 1 pour 2 !

Le nouveau STACKER est arrivé ! version 4.0 Avec cette nouvelle version, vous pouvez désormais transformer une disquette 1,44MB en 3,5MB, plus que doubler la taille de votre disque dur ou multiplier par 2,3 environ la capacité d' une carte mémoire PCMCIA en toute sécurité.
Logiciel à adopter de toute urgence !!! (dispo chez MICROCARDS)

CHOISIR UN POCKET...

par Etienne DEMARCQ

Passionnés de micros de poche et inconditionnels de miniaturisation, notre choix, en 1994, se portera essentiellement sur 4 appareils présents sur le marché français. Nous n'aborderons pas ici les "subnotebooks" et autres "mobile computers" (pardon, Mister ALLGOOD!), qui sont en fait des appareils beaucoup plus volumineux et lourds, et souvent avec une autonomie de fonctionnement réduite à quelques heures. Voici un comparatif ; le plus impartial possible, de ces ordinateurs.

1) Le PORTFOLIO (ATARI)

Apparu en 1990, cet appareil a fait et fait encore le bonheur de milliers de personnes. En son temps, il fût véritablement novateur. Imaginez un compatible PC avec un "grand" écran de 40 colonnes par 8 lignes et 128K RAM de mémoire extensible à 512K. Un slot pour carte mémoire additionnelle et une liaison directe sur PC par l'intermédiaire d'une interface parallèle.

Sur le papier, il semble particulièrement séduisant, cependant, à l'usage on s'aperçoit de quelques vices plutôt désagréables, et quelquefois même franchement détestables. En premier lieu, la ROM interne mis au point par les Anglais D.I.P., dans ses premières versions fut une catastrophe. Les versions 1.030 puis 1.052 et enfin 1.072 étaient bourrées de bugs, avec pour conséquence, des plantages inopinés de la machine au plus mauvais moment (c'est à dire lorsqu'on avait pas encore eu le temps de sauvegarder quoique ce soit !) Heureusement, la version 1.130, impeccable, apparue en 1992, a compensé ses erreurs. Mais le mal était déjà fait. Ensuite, la fameuse compabilité avec le PC n'est qu'assez illusoire, dans la mesure où la taille mémoire est réduite (en version de base), l'écran n'affiche pas la norme 80 par 25 caractères (ce qui oblige à une gymnastique pour faire tourner la plupart des logiciels PC), les interfaces parallèle et série ne sont pas aux normes classiques, et enfin, les cartes mémoires ne sont non plus aux normes internationales PCMCIA. Pour couronner le tout, la société ATARI FRANCE (disparue corps et biens depuis quelques mois), n'a jamais vraiment été à la hauteur sur le plan commercial. Nombreux sont les revendeurs ou les développeurs qui ont baissé les bras devant l'incompétence chronique de cette société, plus habituée à vendre des consoles de jeux que des ordinateurs !

A sa décharge, le PORTFOLIO est un appareil sérieux, solide, relativement puissant pour sa taille, et facile à exploiter. Ses principaux atouts résident dans le fait qu'il possède des logiciels internes qui le transforme en organiseur évolué, à faire pâlir d'envie bien des organiseurs dits de haut de gamme ! Et en plus, c'est un ordinateur ! A 1700F l'appareil se situe à la première place pour le prix, et de loin. Il reste l'ordinateur privilégié des étudiants et gens peu fortunés ou ne désirant pas trop investir. Il a encore de beaux jours devant lui, Le marché de l'occasion se porte très bien.

2) Le HP100LX (Hewlett Packard)

Nous ne parlerons ici que du dernier modèle, le HP100 et non pas du HP95. Pourquoi ? Parcequ'à notre avis, les différences entres les deux machines sont telles que le HP95 a complètement perdu de son intérêt pour quelqu'un qui désire acquérir un nouvel ordinateur. D'ailleurs, il se vend de moins en moins, d'autant plus que le HP100 a baissé de prix. Ce qui ne veut pas dire que le HP95LX soit obsolète, loin de là, mais on s'intéresse ici à une décision éventuelle d'achat de la part du consommateur.

Le HP100LX ou "Cougar" (c'est son nom intime) résulte d'un fabuleux pari technologique. Un écran de 80 par 25lignes, une mémoire ram interne de 1 ou 2 mégas, une compatibilité très correcte avec le monde PC, des logiciels intégrés de très bonne facture de type organiseur, un connecteur de carte PCMCIA de type 2, et une interface série intègrée. Le tout dans un encombrement liliputien et pour un poids très faible. Le dos 5.0 est présent en mémoire morte. Bref, tout pout plaire !

Eh bien, au risque de passer pour un oiseau de mauvais augure, je dirai qu'il y a quelques lacunes ou mauvaises idées conceptuelles concernant cet appareil.

En premier lieu, on est frappé par la taille très petite des touches du clavier. Evidemment pourrait-on rétorquer, avec la taille générale de l'appareil, les ingénieurs de chez HP n'ont pas pu faire mieux. Peut-être, mais s'ils avaient tout simplement augmenté cette taille générale de quelques 1 ou 1,5 cm, cela n'aurait pas beaucoup pénalisé le HP, et le

CHOISIR UN POCKET...

tour était joué. Ce qui aggrave le cas, c'est qu'il possède un clavier numérique séparé (ce qui est une bonne chose en soi, mais cela le pénalise encore plus). Vous me direz que certains utilisateurs ne trouvent pas cela trop gênant, mais ceux qui ont l'intention de beaucoup taper au clavier, la désillusion pourrait venir rapidement. Par contre, dans le cas de l'utilisateur programmeur avant tout, cela présente beaucoup moins d'invénients. Deuxièmement, l'écran. Bien qu'il soit d'un contraste excellent et très lisible, les caractères sont malheureusement beaucoup trop petits ; ils sont minuscules. Le HP100 est à déconseiller aux gens qui ont des problèmes de vue de près. On pourrait rétorquer qu'il possède une fonction zoom qui permet un affichage 64 ou 40 caractères. D'accord, mais on perd alors l'intérêt du 80 colonnes par 25 lignes. Il est dommage également qu'il n'y ait pas de deuxième lecteur de carte mémoire PCMCIA ; ce qui aurait eu pour effet d'augmenter considérablement la puissance et la versatilité de l'appareil. Troisièmement, il ne possède pas d'interface parallèle, et le problème de la connextion à une imprimante se pose alors ; il y a bien la solution du convertisseur série-parallèle, mais cela augmente encore le coût final. Enfin, il ne possède pas de véritable traitement de textes intègré, mais plutôt un éditeur de textes.

Cependant c'est un appareil HP, avec tout le sérieux de la fabrication et du service après vente. De plus, il possède une puissance de calcul phénoménale ! et ce, associée maintenant avec la toute nouvelle version 2MB de mémoire ram. Le tableur est un modèle du genre avec la version Lotus 1-2-3 en mémorte morte. Des périphériques arrivent de plus en plus nombreux également (Communications - Fax- cartes Flasheprom- Interface SCSI- cartes sons etc...) Les matheux, les ingénieurs, trouveront là le pocket (presqu') idéal. Reste le problème du prix et bien que celui-ci ait chuté de 1000F environ, il n'est pas encore adapté à toutes les bourses. Enfin, force est de constater l'extraordinaire engouement qu'il suscite aux USA. Il faut dire que là bas, les tarifs sont beaucoup plus allèchants qu'en Europe.

3) Le PSION Series 3a

Comme pour le HP, nous ne parlerons pas du Psion Series 3, sorti voici déjà quelques années, tant le nouveau Series 3a nous apparaît comme supérieur.

Il s'agit ici d'un appareil très évolué et parfaitement bien pensé. Mais, fidèle à notre habitude, commençons par les reproches.

La principale, et elle est de taille, c'est son incompatibilité avec le monde PC. En effet, le système d'exploitation est un système "maison". Ceci est une attitude typiquement anglaise ! Alors on se met à rêver d'un même appareil avec MS DOS intègré !!!

Ceci étant, la communication des fichiers est possible avec les PC ou les Macintosh. Mais comme il est fortement regrettable de ne pouvoir accéder à tout l'environnement PC ou Mac (les milliers de logiciels souvent gratuits que l'on peut télédécharger via le réseau minitel. L'utilisateur est confiné dans un ensemble de périphériques spéciaux et d'une bibliothèque de logiciels très réduite et souvent onéreuse. Il existe bien quelques "Free-ware"(principalement en Angleterre), mais de toute façon, ceux-ci ne sont pratiquement jamais traduits. Quel dommage pour un ordinateur de cette classe ! Seconde remarque concernant les périphériques. Le Series 3a ne possède ni interface parallèle, ni interface série ; celles-ci sont en option, et les options coûtent très cher ! environ 800F à ajouter si on veut communiquer par la série et 500F si l'on veut imprimer par la parallèle. Quant aux cartes mémoire, les prix sont plus du double des prix pratiqués dans le domaine PCMCIA. Une 1méga ram à 3500F !!! Le futur acheteur devra bien réfléchir au coût de revient final.

Sa mémoire interne de 256K ou 512K, si elle suffit à la majorité des applications internes, est sans aucun doute inadaptée au fameux synthétiseur vocal qui a besoin de 8k par seconde d'enregistrement ! faites le calcul...

Au positif, ce pocket est exceptionnel dans son fonctionnement interne. L'interface utilisateur est d'une simplicité enfantine. L'emploi des touches tactiles est très rapide. Le traitement de texte est formidable, la gestion de fichiers particulièrement efficace, et d'une manière générale toutes les applications le sont aussi. L'écran, généreux, de 79 colonnes par 17 lignes (42 par 11 en traitement de texte) est très lisible et contrasté. Le clavier est parfait au vu de sa petite taille. (n'espérez pas cependant taper avec les 10 doigts)

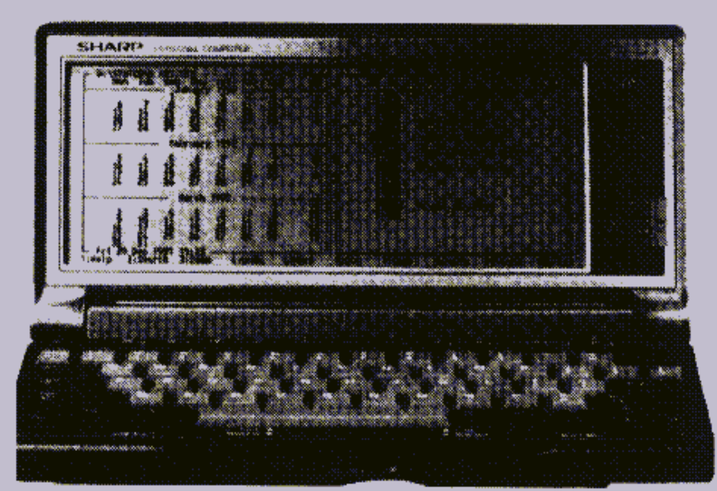
Le système est multi-tâches ; ce qui veut dire que vous pouvez effectuer plusieurs opérations différentes en même temps : étonnant pour un si petit ordinateur !

En résumé, le Psion series 3a conviendra parfaitement aux personnes désirant un super-organiser de poche, ultra puissant et convivial, avec une possibilité intéressante de programmation, mais sans aucune possibilité de réelle connextion au monde PC, un peu meilleure au monde Mac.

4) Le SHARP PC3000

Apparu fin 1992, le Sharp PC3000 a eu une histoire mouvementée. En effet, à l'époque commercialisé par la FNAC et quelques autres revendeurs, il était proposé à quelques 7000F

CHOISIR UN POCKET...



!!! en version de base. La concurrence très rude a eu raison de sa commercialisation en FRANCE. De plus, une erreur dans la rom interne s'était glissée dans les premières versions ; il n'en fallait pas plus pour descendre en flèche cet appareil, au demeurant extraordinairement puissant et bien pensé. En 1993, la société MICROCARDS a relancé la vente en important directement ce produit de l'étranger, et en divisant les prix par deux. Il faut savoir que le Sharp se vend très bien dans les autres pays d'Europe dans le commerce traditionnel. L'erreur dans la rom interne a été corrigé et une nouvelle version est proposée maintenant, depuis septembre 1993, sans aucun problème de fonctionnement.
Bien sûr, les inconditionnels des ordinateurs à taille extrêmement réduites seront déçus. Le SHARP PC3000 ne peut se glisser dans une poche de veste, mais plutôt dans celle d'un imperméable. De plus l'interface homme-machine (sans être compliquée) est beaucoup moins conviviale que le Series3a ou le HP100, en effet, ici c'est le bon vieux système DOS qui est en vigueur, et certains utilisateurs débutants trouveront cela peu pratique. Disons que le SHARP est au système DOS ce que le PSION et le HP sont à WINDOWS au plan du fonctionnement. L'un et l'autre ont leurs fidèles défenseurs ou détracteurs.
Quant à ses qualités, il n'en manque pas, et sa taille mémoire importante permet l'accès à de nombreux logiciels sous DOS. Les deux connecteurs PCMCIA acceptent des cartes RAM, ROM ou Flash Eprom. Son clavier est sensationnel. Agréable, précis et rapide, la largeur de ses touches permet une frappe à 10 doigts. L'écran est un vrai 80 colonnes par 25 lignes avec un bon contraste. L'amateur de textes à taper, l'écrivain, le journaliste, ou tout simplement celui qui désire un certain confort de frappe et de lecture, trouvera là son bonheur. Le programmeur sera enchanté avec toutes les possibilités de programmation qu'il procure : Pascal - Basic - C - Assembleur - etc... et l'électronicien remarquera le bus de type AT standard ainsi que son processeur 80C88 à 10 méga hertz. Il faut noter aussi les 8 logiciels intégrés, et la possibilité très importante de configurer le système, mémoire, langue, clavier, disques internes, etc...
En conclusion, un appareil qui satisfera de nombreux utilisateurs exigeants.

Vous trouverez, ci-après deux tableaux comparatifs des différents ordinateurs de poche ici présentés. Le premier concernant les caractéristiques de chaque appareil, et le deuxième, un essai d'annotation en fonction de critères qui nous apparaissent importants. Le résultat de cette comparaison n'a pour but que de vous informer sur ces produits et de développer votre sens critique en la matière.

Nom	PORTFOLIO	HP95LX	HP100LX	PSION series3	PSION series3a	SHARP PC3000
Prix de vente approximatif	1700F/2500F 128K/512K	2990F/3990F 512K/1MB	4890F/5990F 1MB/2MB	1690F/1990F 128K/256K	3450F/3990F 256K/512K	3450F/3990F/5690F 1MB/2MB/4MB
Dimensions en cm	20x10x2,5	16x8,6x2,5	16x8,6x2,5	16,5x8,5x2,2	16,5x8,5x2,2	22x11x2,5
poids (avec piles)	490g	320g	320g	270g	270g	480g
écran (caractères)	40x8	40x16	80x25	40x8	79x17	80x25
clavier Azerty	oui	oui	oui	oui	oui	oui
clavier chiffres séparés	non	oui	oui	non	non	non
autonomie	30heures	60heures	60heures	80heures	80heures	40heures
microprocesseur	80C88	Nec V20H	80C186	NEC V30H	NEC V36H	80C88
vitesse	4,91MHz	5,33MHz	7,91MHz	3,84MHz	7,68MHz	10MHz
système d'exploitation	MS DOS 2.11	MS DOS 3.22	MS DOS 5.0	propriétaire	propriétaire	MS DOS 3.3
interface parallèle	option	option	option	option	option	oui
interface série	option	option	option	option	option	oui
compatibilité PC	oui	oui	oui	non	non	oui
compatibilité Mac	fichiers	fichiers	fichiers	fichiers	fichiers	fichiers
cartes PCMCIA	0	1 de type 1	1 de type 2	0	0	2 de type 1
cartes spécifiques	1	0	0	2	2	0
traitement de textes	éditeur	éditeur	éditeur	oui	oui	oui
tableur	oui	oui	oui	oui	oui	oui
agenda	oui	oui	oui	oui	oui	oui
base de données	option	oui	oui	oui	oui	oui
capacité mémoire maxi	512K	1MB	2MB	256K	512K	4MB

CHOISIR UN POCKET...

Nom	PORTFOLIO	HP95LX	HP100LX	PSION series3	PSION series3a	SHARP PC3000
taille et encombrement	15	20	20	20	20	13
poids	15	19	19	20	20	16
écran	8	10	14	9	16	18
clavier	17	11	11	16	16	20
autonomie	12	16	17	18	20	15
interface utilisateur	13	15	17	18	20	14
compatibilité PC	10	12	18	6	6	18
compatibilité Mac	6	6	6	10	10	6
périphériques	9	12	18	7	9	17
traitement de textes	12	10	13	16	18	17
tableur	15	18	20	0	17	17
agenda	15	15	18	20	20	18
base de données	0	14	18	19	19	17
capacité mémoire	11	13	18	6	12	19
pérénité et service après vente	12	14	18	12	18	18
prix	20	18	14	20	17	17
Total des notes	184	218	259	215	258	260
Classement	6ème	4ème	2ème	5ème	3ème	1er

Commentaires :

Ce tableau permet un classement approximatif des pockets les plus connus en FRANCE. Vous remarquerez que les appareils les plus modernes arrivent en tête ; c'est normal ! Cependant, les écarts entre eux ne sont pas significatifs, les 3 premiers se situent dans un mouchoir de poche (c'est le cas de le dire). On peut considérer que l'acheteur orientera son choix essentiellement en fonction de ses besoins. S'il désire la puissance et la plus parfaite compatibilité et le besoin de frapper beaucoup de textes, il choisira sans hésitations le SHARP PC3000. Au contraire, si pour lui, il doit effectuer beaucoup de calculs et pouvoir se connecter à des PC pour utiliser leur tableur, il choisira le HP100LX. Enfin, s'il est séduit par la petitesse et la grande convivialité, et que l'accès au monde PC l'importe peu, il choisira le PSION series3a. A ce niveau également, le prix peut avoir une influence, mais ne doit pas être le seul critère de sélection. Bien sûr, s'agissant de pockets, la taille est très importante, toutefois, attention à ne pas tomber dans l'extrême inverse, la grosseur de vos doigts, elle, ne change pas ! Interrogez-vous également sur les possibilités de connextion et sur la bibliothèque de logiciels avant de vous décider. Enfin, il est des critères dits irrationnels qui vont vous faire sélectionner tel ou tel appareil simplement par impulsion. Et quelquefois, ce n'est pas si mal !

JEU DE CASES SUR PORTFOLIO

par Thierry BODIN

Voici un petit jeu fort intéressant pour le PORTFOLIO. Ecrit en basic, il permettra à de nombreux amateurs de s'initier à ce langage.

```
100 randomize
dim tabl(7,7),tabl2(7,7),cp$(8)
200 cls
open»i»,#1,»memogame.scr»
input #1,rcr
close #1
locate 1,9:print»Thierry BODIN Présente:»
box 3,10,5,29,1 locate 4,12,0
print»Mémo-Game v1.0 !»
locate 7,14:print»Aide (O/N) ?»
22 a$=inkey$
if a$=»» then 22
if a$=»o» or a$=»O» then gosub 20
locate 7,11:print»Initialisation ...»
for i=1 to 7
cp$(i)=»  «
for j=1 to 7
tabl(i,j)=219
tabl2(i,j)=0
next j:next i
cp$(8)=»  «
gosub 1
tabl2(x,y)=asc(«%»)
for i=1 to 24:j=1
locate 7,29:print int(((i-1)*2+1)*100/48);
print»%»
gosub 1
tabl2(x,y)=i+64
gosub 1
tabl2(x,y)=i+64
next i
cls
gosub 2
box 1,23,5,38,0
score=0
nbc=0
locate 8,25:print»Record: «;:print rcr;:locate
1,1
3 c1$=»  «:c2$=»  «
for i=1 to 8
locate i,19:print»¦»+cp$(i)+»¦»;:locate 1,1
next i
locate 2,25:print»Case 1: «+c1$+»  «
locate 3,25:print»Case 2: «+c2$+»  «
locate 4,25:print»Score :     «
locate 4,33:print score
locate 2,33,2:input «»,c1$
if c1$=»0" then cls:end
if len(c1$)<2 then 3
x1=asc(mid$(c1$,1,1))-64
if x1>32 then x1=x1-32
y1=asc(mid$(c1$,2,1))-48
if x1<1 or x1>7 then 3
if y1<1 or y1>7 then 3
if tabl(x1,y1)<>219 then 3
gosub 75
cp$(8)=C1$
locate y1+1,x1*2+1:print
chr$(tabl2(x1,y1));:locate 1,1
if chr$(tabl2(x1,y1))=»%» then
tabl(x1,y1)=asc(«%»):goto 10
4 locate 3,33:print»  «:C2$=»  «
locate 3,33:input «»,c2$
if len(c2$)<2 then 4
x2=asc(mid$(c2$,1,1))-64
if x2>32 then x2=x2-32
y2=asc(mid$(c2$,2,1))-48
if x2<1 or x2>7 then 4
if y2<1 or y2>7 then 4
if tabl(x2,y2)<>219 then 4
if c2$=c1$ then 4
gosub 75
cp$(8)=c2$
locate y2+1,x2*2+1:print
chr$(tabl2(x2,y2));:locate 1,1
if chr$(tabl2(x2,y2))=»%»
then tabl(x2,y2)=asc(«%»):locate
y1+1,x1*2+1:print»_»;:locate 1,1:goto 10
if tabl2(x2,y2)=tabl2(x1,y1) then 11
locate 7,25,0:print»Loupé !»
for i=1 to 200:next i
locate 7,25,0:print»      «
locate y1+1,x1*2+1:print»_»;
locate y2+1,x2*2+1:print»_»;:locate 1,1,0
score=score-1:goto 3
10 locate 7,25,0:print»BONUS
!»:score=score+50:gosub 40:wait:goto 3
11 tabl(x1,y1)=tabl2(x1,y1)
tabl(x2,y2)=tabl2(x2,y2)
nbc=nbc+1:locate 7,25,0:print»
Bien !»:score=score+2
gosub 40
for i=1 to 200:next i:locate 7,25,0:print»
«
if nbc=24 then locate 7,25,0:print»Fini
!»:gosub 40:wait:goto 123
goto 3
2 locate 1,1,2:print»\ A B C D E F G»
for i=1 to 7
locate i+1,1:print
i;:print»_____»;:locate 1,1
for j=1 to 7
locate i+1,j*2+1:print chr$(tabl(i,j));:locate
1,1
next j
next i
return
1 x=int(rand(7)+1)
y=int(rand(7)+1)
if tabl2(x,y)>0 then 1
return
75 for i=2 to 8
cp$(i-1)=cp$(i)
next i:cp$(8)=»  «:return
20 cls
print»Le but du jeu est de trouver les cases»
print»identiques dans la  grille  ou bien le»
print»bonus représenté par '%'.»
print»Exemple de jeu: Vous tapez B4  puis
F6"
print» Ces cases contiennent 'F' et 'P'.vous»
print» perdez 1 point. Vous jouez ensuite
C7"
gosub 21
print» qui contient 'P'. Tapez alors  F6 qui»
print» contient aussi 'P'. Vous gagnez 2 pts»
print» et les 2 cases restes affichées.»
print» Pour arrêter tapez 0 dans la case 1.»
print:print»Bon amusement !»
gosub 21:return
21 locate 8,1,0:print»Pressez une
touche...»;:locate 1,1,0:wait:cls:return
40 sound 35,8:sound 39,8:sound 45,8:return
123 if score>rcr then 124
goto 200
124 cls
locate 3,10:print»Nouveau Record !»;:print
score
open»O»,#1,»memogame.scr»
print #1,score
close #1
wait
goto 200
```

PDRAW POUR PORTFOLIO

par M. BARRY

Pdraw 2.0 est un logiciel de dessin très évolué tournant sur Portfolio standard. Entièrement écrit en PBASIC 4.5 (exportable sur PC), il démontrera que ce langage n'est pas un sous langage, et que, bien maîtrisé, il permet de faire beaucoup de choses... Cet article a aussi un aspect didactique, dans la mesure ou de nombreux débutants pourront comprendre et s'exercer à la programmation.

```
Initialisations
  on error goto 30
  if indfd=1 then indfd=0 goto 37
  if indtr=1 then gosub 93: indtr=0 goto 85
  if indret=1 then 21
  if pass=1 then 20
  pass=1
  fihelp$=»pdraw.hlp»
  fitemp$=»pdraw.tmp»
  filogo$=»pdraw.lgo»
  fichhlp$=»pdrawhlp.bas»
  fichfon$=»pdrawfn1.bas»
  fichnua$=»pdrawnua.bas»
  fichdiv$=»pdrawdiv.bas»
  fichico$=»pdrawico.bas»
  fichtra$=»pdrawtra.bas»
  fiwa$=»1": screen 4
  pgload filogo$
  wait: cls: gosub 10: goto 25
10 pgsave fitemp$: cls
  x=120: y=32
13 pixel=point(x,y)
  pset(x,y),pixel xor 1
  c=1: pas=15: I=0: J=1
  gosub 27: swap c,pas: pol$=»1"
  s=0: l=0: n=0: pp=0: qq=1
  mode$=»curseur»: pst=1: psc=1
  return
'**** Traitement clavier : commandes
20 a$=inkey$: if a$=»» then 20
21 indret=0
  if a$=»K» then 40  ' Gauche
  if a$=»M» then 50  ' Droite
  if a$=»H» then 60  ' Haut
  if a$=»P» then 70  ' Bas
  if a$=»G» then 41  'H/G
  if a$=»I» then 51  'H/D
  if a$=»Q» then 61  'B/D
  if a$=»O» then 71  'B/G
   if a$=»x» then pst=1:psc=1:w=0:v=1:goto 24
'mode curseur
  indpt=0
  if a$=»g» then mode$=»gomme» :gosub
 22:pst=0:psc=0:pix3=0 goto 20  'mode gomme
  if a$=» « then mode$=»crayon»:gosub
 22:pst=1:psc=1:pix3=1 goto 20  'mode crayon
   if a$=chr$(27) then gosub 22: gosub 35:screen 7
goto 31        'retour
  if a$=»w» then indpt=1 goto 20     'dépot 1 pixel
  if a$=»i» then gosub 35: screen 7: indint=1: chain
fichdiv$   'longueur pas
  if a$=»l» then gosub 22: gosub 35: n=0: screen 7
goto 90 'lire image
  if a$=»s» then gosub 22: gosub 35: n=0: screen 7
goto 91 'sauvegarder
  if a$=»:» then swap c,pas goto 20
 'échanger pas
   if a$=»e» then kill fitemp$: gosub 10 goto 26
'effacer ecran
  if a$=»d» then x1=x:y1=y:indd=1:indv=0 goto 20
'depart
  if a$=»t» then 100                'trait
  if a$=chr$(20) then indis=1: goto 100  'trait invvid
  if a$=»q» then 200                'quadrilatere
   if a$=chr$(16) then indis=1: goto 200    'quad.
invvid
  if a$=»c» then 400                'cercle
  if a$=»a» then 404                'arc
  if a$=»o» then 410                'ovale
  if a$=»r» then 82
'remplissage trames
   if a$=»m» then gosub 35: screen 7: indmem=1:
chain fichdiv$ 'mémo
  if a$=»k» then gosub 22: gosub 35: screen 7: chain
fichfon$ 'caractères
  if a$=»%» then gosub 22: gosub 35: chain fichfonS
'caractères
  if a$=»h» then gosub 35: screen 7: chain fichhlp$
'aide en ligne
  if a$=»f» then indm=0: ind2=0 goto 700
   'déf. zone pour copie
  if a$=»v» or a$=»/» then 701
 'copie de zone
  if a$=»z» then 800            'effacement zone
  if a$=»n» then 900            'inversion video
  if a$=»j» then gosub 22: gosub 35: screen 7: chain
fichnua$       'nuage points
  if a$=»$» then gosub 22: gosub 35: chain fichnua$
'nuage points
   if a$=»y» then gosub 22: gosub 35: indicon=0:
chain fichico$       'création icone
   if a$=»u» then gosub 22: gosub 35: screen 7:
indicon=1: chain fichico$  'incrustation icone
   if a$=»b» then gosub 22: gosub 35: screen 7:
indfd=1 chain fichdiv$ 'trames de fond
  if a$=»p» then gosub 22: gosub 35: gosub 88: chain
fichtra$
  if a$=»1" or a$=»2" or a$=»3" then pol$=a$: click
goto 20        'valeur de n
  beep: goto 20
22 if pp=1 then swap pp,qq: gosub 23
  return
24 if mode$=»crayon» then pixel=1
 mode$=»curseur» goto 25
  if mode$=»gomme»  then pixel=0
 mode$=»curseur»
25 swap pp,qq
  if pp=1 then gosub 23 goto 20
  pset(x-1,y-1),pixel1
  pset(x-1,y+1),pixel2
  pset(x+1,y-1),pixel3
  pset(x+1,y+1),pixel4
  goto 20
26 swap pp,qq: gosub 23
28 a$=inkey$: if a$=»» then 28
  if a$<>»z» then goto 21
  pgload fitemp$
  goto 20
27 indd=0:indv=0:indm=0:ind2=0:inds=0
   indfmz=0:indnm=0:indt=0:indu=0
   indal=0:indg=0:indf=0:inda=0:indsw=0
   indext=0:indpass=0:indn=0
  return
23 pixel1=point(x-1,y-1)
  pset(x-1,y-1),pixel1 xor 1
  pixel2=point(x-1,y+1)
  pset(x-1,y+1),pixel2 xor 1
  pixel3=point(x+1,y-1)
  pset(x+1,y-1),pixel3 xor 1
  pixel4=point(x+1,y+1)
  pset(x+1,y+1),pixel4 xor 1
  return
'**** Traitement «on error goto...»
30 if n=1 then n=0: screen 4: pgload fitemp$: resume
99
  if s=1 then s=0: resume 33
  if l<>1 then screen 4: beep:beep resume 20
  l=0: screen 7
  print «  ***** PDRAW - Lecture image «
  print « «
  print «  * Fichier image non trouvé «
  wait: cls: resume 90
'****  Effacement fichier et sauvegarde
32 n=0: fiw$=fiw$+».pgc»
  s=1: kill fiw$
33 screen 4: pgload fitemp$: pgsave fiw$
  fisav$=fiw$
  s=0: click: return
'****  Sauvegarde temporaire image
35 n=1: kill fitemp$: pgsave fitemp$ return
'****  Sortie de programme
31 print «  ***** PDRAW - Sortie programme «
  print « «
  input «  * Nom de l'image : «,fiw$
  if fiw$=»=» then cls: system
   if fiw$=»+» then screen 4: pgload fitemp$: goto
20
  if fiw$<>»» then gosub 32: screen 2: system
  fiw$=fiwa$
  if fiw$=»1" then beep: cls: goto 31
  gosub 32
  screen 2: system
37 if indz=1 then indz=0 goto 20
  if indno=1 then indno=0 goto 20
  goto 21
'****  Point courant : dépl. G
40 if mode$<>»curseur» then goto 42
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  x=x-c
  if x<0 then x=0
44 if pp=1 then 46
  pixel=point(x,y)
  pset(x,y),pixel xor 1
  indpt=0: goto 20
'**** Point courant : dépl. H/G
41 if mode$<>»curseur» then 43
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  x=x-c: y=y-c
  if x<0 then x=0
  if y<0 then y=0
  goto 44
'**** Point courant : dépl. H/D
51 if mode$<>»curseur» then 53
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  x=x+c: y=y-c
  if x>239 then x=239
  if y<0  then y=0
  goto 44
'**** Point courant : dépl. B/D
61 if mode$<>»curseur» then 63
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  x=x+c: y=y+c
  if x>239 then x=239
  if y>63  then y=63
  goto 44
'**** Point courant : dépl. B/G
71 if mode$<>»curseur» then 73
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  x=x-c: y=y+c
  if x<0  then x=0
  if y>63 then y=63
  goto 44
```

PDRAW POUR PORTFOLIO

```
43 if mode$=»gomme» then pset(x,y),0
  x=x-c: y=y-c
  if x<0 then x=0
  if y<0 then y=0
  gosub 47 goto 20
53 if mode$=»gomme» then pset(x,y),0
  x=x+c: y=y-c
  if x>239 then x=239
  if y<0  then y=0
  gosub 47 goto 20
63 if mode$=»gomme» then pset(x,y),0
  x=x+c: y=y+c
  if x>239 then x=239
  if y>63  then y=63
  gosub 47 goto 20
73 if mode$=»gomme» then pset(x,y),0
  x=x-c: y=y+c
  if x<0  then x=0
  if y>63 then y=63
  gosub 47 goto 20
45 pset(x-1,y-1),pixel1
  pset(x,y),pixel
  pset(x-1,y+1),pixel2
  pset(x+1,y-1),pixel3
  pset(x+1,y+1),pixel4
  indpt=0: return
46 pixel1=point(x-1,y-1)
  pset(x-1,y-1),pixel1 xor 1
  pixel=point(x,y)
  pset(x,y),pixel xor 1
  pixel2=point(x-1,y+1)
  pset(x-1,y+1),pixel2 xor 1
  pixel3=point(x+1,y-1)
  pset(x+1,y-1),pixel3 xor 1
  pixel4=point(x+1,y+1)
  pset(x+1,y+1),pixel4 xor 1
  goto 20
47 if mode$=»gomme» then pset(x,y),1:goto 49
  pset(x,y),pix3
49 if pol$=»1" then return
  pset(x-1,y),pix3: pset(x,y-1),pix3
  pset(x-1,y-1),pix3
  if pol$=»2" then return
  pset(x+1,y-1),pix3: pset(x+1,y),pix3
  pset(x+1,y+1),pix3: pset(x,y+1),pix3
  pset(x-1,y+1),pix3 return
'**** Point courant : dépl. D
50 if mode$<>»curseur» then goto 52
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  x=x+c
  if x>239 then x=239
  goto 44
'**** Point courant : dépl. H
60 if mode$<>»curseur» then goto 62
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  y=y-c
  if y<0 then y=0
  goto 44
'**** Point courant : dépl. B
70 if mode$<>»curseur» then goto 72
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
  y=y+c
  if y>63 then y=63
  goto 44
42 if mode$=»gomme» then pset(x,y),0
  x=x-c
  if x<0 then x=0
  gosub 47 goto 20
52 if mode$=»gomme» then pset(x,y),0
  x=x+c
  if x>239 then x=239
  gosub 47 goto 20
62 if mode$=»gomme» then pset(x,y),0
  y=y-c
  if y<0 then y=0
  gosub 47 goto 20
72 if mode$=»gomme» then pset(x,y),0
  y=y+c
  if y>63 then y=63
  gosub 47 goto 20
80 if pp=1 then swap pp,qq: gosub 23
  gosub 35: gosub 88: return
88 x2=x: y2=y: z1=x1: z2=y1: z3=x2: z4=y2 return
85 click: indf=0: indd=0: indis=0: n=0
  if mode$=»gomme» then goto 99
83 z$=inkey$ : if z$=»» then goto 83
  mode$=»curseur»: x=z3: y=z4
  if z$=»z» then pixel=0: pgload fitemp$ goto 20
  if inds=1 then inds=0 a$=z$ goto 21
  if indfmz=1 then indfmz=0:pixel=0 else pixel=1
  a$=z$ goto 21
'**** Remplissage Trames
82 if pp=1 then swap pp,qq: gosub 23
  gosub 35: x1=x:y1=y
  gosub 88: gosub 300
  n=0: indnm=0: indt=0: indu=0: indal=0
  indg=0
  click: goto 83
'**** Lire une image sur disque
90 print «  ***** PDRAW - Lecture image «
  print « «
  input «  * Nom de l'image : «,fir$ : screen 4
  if fir$=»» then pgload fitemp$ goto 20
  l=1
  pgload fir$: filec$=fir$
  gosub 13 goto 28
'**** Sauvegarder une image sur disque
91 print «  ***** PDRAW - Sauvegarde image «
  print « «
  input «  * Nom de l'image : «,fiw$
  if fiw$=»=» then screen 4: pgload fitemp$ goto 20
  if fiw$<>»» then fiwa$=fiw$:gosub 32 goto 20
  fiw$=fiwa$
  if fiw$=»1" then beep: cls: goto 91
  gosub 32 goto 20
99 mode$=»curseur»
  pst=1: psc=1
  gosub 93: goto 20
93 x=z3: y=z4: pixel=0
  pset(x,y),1
  gosub 27: indfmz=1: n=0
  return
'**** Fonction TRAIT
'**** Trait : horizontal/vertical/oblique=45°
100 if indd=0 then beep: goto 20
  gosub 80
  if x1=x2 and y1=y2 then beep: goto 20
  nb=val(pol$): aa=0: pix=0
  if x1<x2 and y1=y2 then 101
  if x1>x2 and y1=y2 then swap x1,x2 goto 101
  if x1=x2 and y1<y2 then 103
  if x1=x2 and y1>y2 then swap y1,y2 goto 103
  x=x2-x1: y=y2-y1
  a=abs(x): b=abs(y)
  if a=b then 105 else goto 111
101 for xx=x1 to x2
    if indis=1 then pix=point(xx,y1+aa)
    pset(xx,y1+aa),pst xor pix
  next xx
  if nb=1 then 85
  if nb=2 then aa=-1: nb=1 goto 101
  aa=1: nb=2 goto 101
103 for yy=y1 to y2
    if indis=1 then pix=point(x1+aa,yy)
    pset(x1+aa,yy),pst xor pix
  next yy
  if nb=1 then 85
  if nb=2 then aa=-1: nb=1 goto 103
  aa=1: nb=2 goto 103
105 if x1<x2 and y1>y2 then swap x1,x2: swap y1,y2
goto 110
  if x1<x2 and y1<y2 then swap x1,x2: swap y1,y2
goto 109
  if x1>X2 and y1>y2 then 109
110  yy=0
    for xx=x2 to x1
     if indis=1 then pix=point(xx+aa,y2-yy)
     pset(xx+aa,y2-yy),pst xor pix
     incr(yy)
   next xx
   if nb=1  then 85
   if nb=2  then aa=1 : nb=21:decr(x1) goto 110
   if nb=21 then aa=-1: nb=1: decr(y2)
:incr(x2):incr(x1) goto 110
   if nb=3  then aa=1 : nb=31:decr(x1) goto 110
   if nb=31 then aa=-1: nb=32:decr(y2)
:incr(x2):incr(x1) goto 110
   if nb=32 then aa=2 : nb=33:x1=x1-2:incr(y2):
decr(x2):goto 110
  aa=-2: nb=1: y2=y2-2:x2=x2+2:x1=x1+2 goto 110
109 yy=0
    for xx=x2 to x1
     if indis=1 then pix=point(xx+aa,y2+yy)
     pset(xx+aa,y2+yy),pst xor pix
     incr(yy)
   next xx
  if nb=1  then 85
  if nb=2  then aa=1 : nb=21:decr(x1) goto 109
        if    nb=21    then    aa=-1:    nb=1
:incr(y2):incr(x2):incr(x1) goto 109
   if nb=3  then aa=1 : nb=31:decr(x1) goto 109
if nb=31 then aa=-1: nb=32:incr(y2):incr(x2):incr(x1)
goto 109
if nb=32 then aa=2 : nb=33:x1=x1-2:
decr(y2):decr(x2) goto 109
  aa=-2: nb=1: y2=y2+2: x2=x2+2: x1=x1+2 goto
109
'**** Trait : oblique<>45°
111 fint=0:inda=0:pix=0
   x=x2-x1:y=y2-y1
   if x<0 then swap x1,x2:swap y1,y2 goto 111
   A=abs(x):B=abs(y)
   if y<0 then goto 112
   if A>B then goto 113
 swap A,B:swap x1,y1:swap x2,y2:indsw=1 goto 113
112 if  A<B  then  inda=1:Q=B\A:Q1=Q:R=B-
(Q*A):gosub 150 goto 115
   Q=A\B:Q1=Q:R=A-(Q*B): gosub 150
114 gosub 180 :gosub 130
   if fint=1 then 85
   if Q1<>0 then incr(x1) goto 114
   Q1=Q
   if indext=1 then goto 125
126 if y1=y2 then incr(x1) goto 103 else decr
(y1):incr(x1) goto 114
113 Q=A\B:Q1=Q:R=A-(Q*B): gosub 150
122 if indsw=1 then gosub 181 else gosub 180
   gosub 130
   if fint=1 then 85
   if Q1<>0 then incr(x1) goto 122
   Q1=Q
   if indext=1 then goto 145
146 if y1=y2 then incr(x1) goto 122
   incr(y1): incr(x1) goto 122
145 gosub 170
   if fint=1 then 85 else goto 146
125 gosub 170
   if fint=1 then 85 else goto 126
115 gosub 180: gosub 130
   if fint=1 then 85
   if Q1<>0 then decr(y1) goto 115
```

PDRAW POUR PORTFOLIO

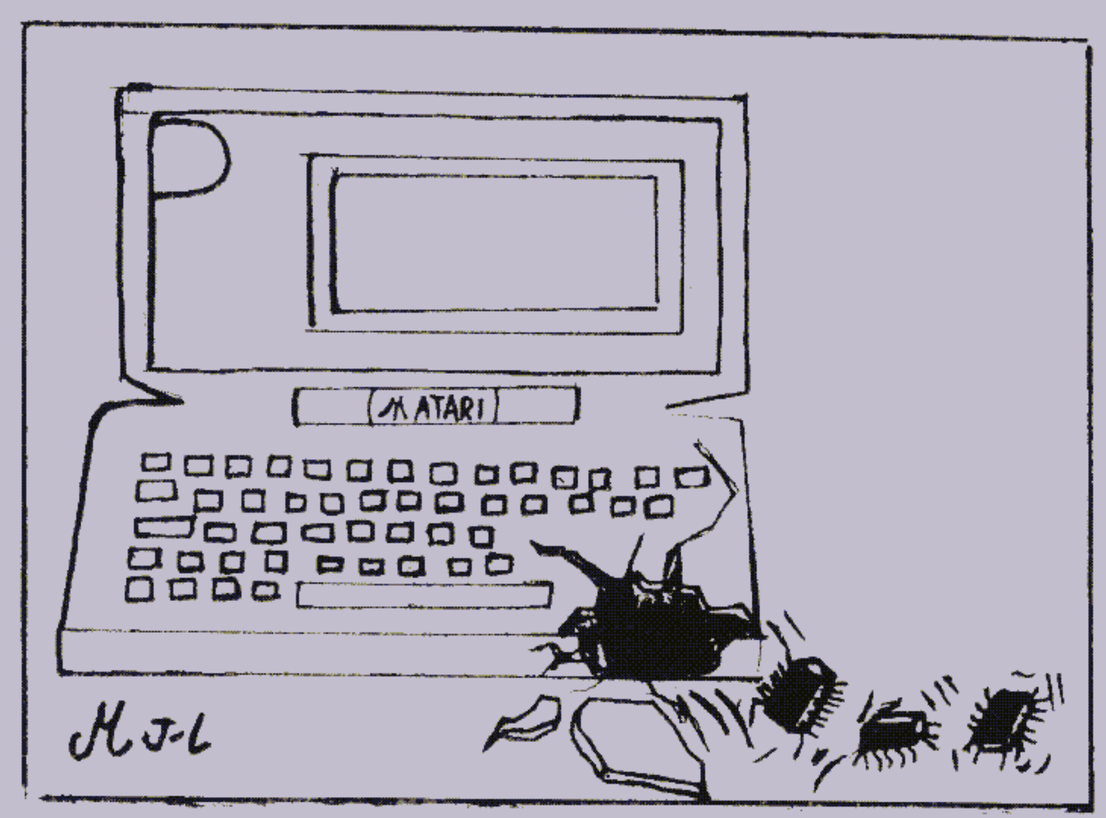
```
  Q1=Q
  if indext=1 then goto 135
136 if x1=x2 then decr(y1) goto 115
  incr(x1): decr(y1) goto 115
135 gosub 170
  if fint=1 then 85 else goto 136
130 if x1<>x2 then decr(Q1) return
  if y1<>y2 then decr(Q1) return
  fint=1: indsw=0
  x=z3: y=z4: return
'**** Trait : calcul des extensions
150 indext=0:D=2  'D=intervalle entre 2 extensions
 if R=0 then return  'R2=reste après 1ères extensions
151 if inda=0 then Q2=B\D:R2=B-(D*Q2) else
Q2=A\D:R2=A-(D*Q2)
  if Q2=0 then return  'Q2=nb de 1ere extensions
  Q3=R\Q2:Q4=Q3  'Q3=nb pixels 1ere extension
  if Q3<1 then incr(D) goto 151
  R3=R-(Q2*Q3)  'R3=nb de 2emes extensions
  indext=1: D1=D: return
'**** Trait : éxécution des extensions
170 decr(D)
  if D<>0 then return
  D=D1
171 if inda<>0 then decr(Y1):gosub 180 goto 172
  incr(X1)
  if indsw=1 then gosub 181 else gosub 180
172 decr(Q3)
  if Q3<>0 then goto 171
  if R3<>0 then goto 174
175 Q3=Q4: decr(Q2)
  if Q2<>0 then return
  indext=0
  return
174 if indpass=0 then indpass=1 goto 175
  indpass=0: decr(R3)
  if inda<>0 then decr(y1): gosub 180 goto 175
  incr(x1)
   if indsw=1 then gosub 181 goto 175 else gosub
180 goto 175
'**** Trait : traçage oblique<>45°
180 if indis=1 then pix=point(x1,y1)
  if nb=1 then pset(x1,y1),pst xor pix return
  x=x1: y=y1: pix3=pst
  gosub 47 return
181 if indis=1 then pix=point(y1,x1)
  if nb=1 then pset(y1,x1),pst xor pix return
  x=y1: y=x1: pix3=pst
  gosub 47 return
'**** Fonction QUADRILATERE
200 if indd=0 then beep: goto 20
  gosub 80
  if x1=x2 and y1=y2 then beep: goto 20
  if x1>x2 then swap x1,x2
  if y1>y2 then swap y1,y2
  pix=0: nb=val(pol$)-1
  if nb=2 then decr(x1):decr(y1):incr(x2):incr(y2)
 for aa=0 to nb
  for xx=x1+aa to x2-aa
    if indis=1 then pix=point(xx,y1+aa)
    pset(xx,y1+aa),pst xor pix
    if indis=1 then pix=point(xx,y2-aa)
    pset(xx,y2-aa),pst xor pix
  next xx
  for yy=y1+1+aa to y2-1-aa
    if indis=1 then pix=point(x1+aa,yy)
    pset(x1+aa,yy),pst xor pix
    if indis=1 then pix=point(x2-aa,yy)
    pset(x2-aa,yy),pst xor pix
  next yy
 next aa
  pix=0: goto 85
'**** Fonction Remplissage trames
300 screen 7
  print « ***** PDRAW - Trames «
  print « «
  input «  * Trame _____ (o/n) : «,rep$
  if rep$=»o» then goto 305
  input «  * Trame _____ (o/n) : «,rep$
  if rep$=»o» then indnm=1 goto 305
  input «  * Trame _____ (o/n) : «,rep$
  if rep$=»o» then indu=1 goto 305
  input «  * Trame ::::: (o/n) : «,rep$
  if rep$=»o» then indnm=1: indt=1 goto 305
  input «  * Trame ::::: (o/n) : «,rep$
 if rep$=»o» then indg=1: indt=1: indnm=1 goto 305
  input «  * Trame aléat (o/n) : «,rep$
  if rep$=»o» then indal=1 goto 305
  screen 4: pgload fitemp$
  inds=1: return
305 screen 4: pgload fitemp$
  a=x: b=y: psn=1: pix=0
  p1=1: p2=0: psm1=1: psm2=0
  gosub 301
302 if indt=0 then swap psm1,psm2
  decr(b)
  if indu=1 then swap p1,p2: psn=p1
  if b<0 then goto 311
  pix=point(a,b)
  if pix=1 then goto 311
  if indg=0 then gosub 301 goto 302
  decr(b)
  if b<0 then goto 311
  pix=point(a,b)
  if pix=0 then gosub 301 goto 302
311 a=x: b=y: p1=1: p2=0
  psm1=1: psm2=0
304 if indt=0 then swap psm1,psm2
  incr(b)
  if indu=1 then swap p1,p2: psn=p1
  if b>63 then return
  pix=point(a,b)
  if pix=1 then return
  if indg=0 then gosub 301 goto 304
  incr(b)
  if b>63 then return
  pix=point(a,b)
  if pix=0 then gosub 301 goto 304
  return
301 n1=psm2: n2=psm1
303 if indnm=1 then swap n1,n2: psn=n1
  if indal=1 then psn=rand(2)
  pset(a,b),psn: a=a-1
  if a<0 then goto 310
  pix=point(a,b)
  if pix=0 then goto 303
310 a=x: n1=psm1: n2=psm2
306 incr(a)
  if a>239 then a=x: return
  if indnm=1 then swap n1,n2: psn=n1
  if indal=1 then psn=rand(2)
  pix=point(a,b)
  if pix=0 then pset(a,b),psn goto 306
  a=x: return
'**** Fonctions ARC, CERCLE, OVALE
400 if indd=0 then beep: goto 20
  gosub 80
  if x1=x2 and y1=y2 then beep: indd=0: goto 20
  angle=360: ellipse=1: reso=8
  angle1=angle: gosub 401: indd=0: n=0: goto 85
405 screen 4: pgload fitemp$
401 if x1=x2 then R=abs(y2-y1):x2=x1+R goto 402
  R=abs(x2-x1)
  if x2<x1 then dep=180:x2=x1+R else dep=0
403 ang=angle: ell=ellipse
  arr=dep+angle
  stv=360/R/reso
  for theta=dep to arr step stv
    angle=theta*3.1415/180
    sx=R*cos(angle)
    sy=R*sin(angle)
    sx=x2+sx-R
    sy=y2+sy*ellipse
    x=sx: y=sy: pix3=psc
    gosub 47
    sx1=sx: sy1=sy
    next theta
  return
402 if y2<y1 then dep=270:y2=y1 goto 403
  dep=90:y2=y1 goto 403
404 if indd=0 then beep: goto 20
  gosub 80
  if x1=x2 and y1=y2 then beep: indd=0: goto 20
  angle=180: ellipse=1: reso=8
  gosub 35: screen 7
  print «  ***** PDRAW - Fonction ARC «
  print « «
  input «  * Valeur de l'angle : «,rep$
  if rep$<>»» then angle=val(rep$)
angle1=angle:gosub                    412:gosub
405:indd=0:n=0:goto 85
410 if indd=0 then beep: goto 20
  gosub 80
  if x1=x2 and y1=y2 then beep: indd=0: goto 20
  angle=360: ellipse=0.5: reso=8
  gosub 35: screen 7 gosub 412
  gosub 405: indd=0: n=0: goto 85
412 if a$=»a» then 413
  print «  ***** PDRAW - Fonction OVAL «
  print « «
413 input «  * Valeur de l'ellipse (.1->3): «,rep$
  if rep$=»» then return
  ellipse=val(rep$)
  if ellipse>2 then reso=20 :return
  if ellipse>1.2 then reso=14: return
  return
'**** Transfert d'image
700 if indd=0 then beep: goto 20
  if x<x1 or y<y1 then beep: goto 20
  x3=x: y3=y: indf=1 goto 20
701 if indf=0 then beep: goto 20
  if pp=1 then swap pp,qq gosub 23
  gosub 88: indtr=1: chain fichdiv$
'**** Effacement d'une zone
800 if indd=0 then beep: goto 20
  gosub 88
  if x2=x1 and y2=y1 then beep: indd=0: goto 20
  gosub 22: gosub 35
  if x1>x2 then swap x1,x2
  if y1>y2 then swap y1,y2
  for yy=y1 to y2
    for xx=x1 to x2
      pset(xx,yy),0
    next xx
  next yy
  gosub 93
  goto 85
'**** Inversion video sur place
900 if indd=0 then beep: goto 20
  gosub 88
  if x2=x1 and y2=y1 then beep: indd=0: goto 20
  gosub 22: gosub 35
  if x1>x2 then swap x1,x2
  if y1>y2 then swap y1,y2
  for yy=y1 to y2
    for xx=x1 to x2
      pix=point(xx,yy)
      pset(xx,yy),pix xor 1
    next xx
  next yy
  gosub 93
  if pix=1 then indfm=0
  goto 85
1001 return
002 return
'004 return
'005 return
'003 return
  end
```

TEST MEMOIRE POUR PORTFOLIO

par G.C

Ce programme écrit en assembleur 8086 permet de tester les cartes mémoire RAM pour Portfolio jusqu'à 1 Megaoctects. Amis programmeurs, accrochez vos ceintures !!!

La mémoire d'une carte PORTFOLIO (support A: ou B:) fait partie intégrante de l'espace adressable du 80C88, de 0C0000H à 0DFFFFH soit un maximun de 128Koctets.
Un artifice appelé pagination permet de dépasser cette taille fatidique, l'écriture d'un zéro dans le registre de page permet d'adresser les 128 premiers Koctets , l'écriture d'une valeur N permet d'atteindre les N x 128 Koctets suivants.
Ce registre de page fait aussi partie de l'espace mémoire de la carte (0C000AH), ce qui rend inutilisable l'octet à cette adresse dans toutes les pages.
Comme il n'est pas possible de tester toutes les configurations d'octets en mémoire, deux types de tests peuvent être utilisés avec de bon résultats :
le 0 ou le 1 baladeur, et le damier bouclant.
La figure ci-dessous donne la configuration des bits sur les octets en mémoire



	1 baladeur	0 baladeur	damier bouclant
Octet N	00000000	11111111	01010101
N+1	00000001	11111110	10101010
N+2	00000010	11111101	01010101
N+3	00000100	11111011	10101010
N+4	00001000	11110111	01010101
N+5	00010000	11101111	
N+6	00100000	11011111	
N+7	01000000	10111111	
N+8	10000000	01111111	
N+9	00000000	11111111	

On remarque tout de suite que le motif des 0 ou des 1 baladeurs se reproduit tous les 9 octets tandis que le damier ne possède que deux valeurs possibles ; (55H et 0AAH).
Dans l'exemple ci-dessus la valeur de départ pour l'octet N est de 00H, 0FFH ou 55H selon le type de test, rien n'empêche d'en changer à chaque passe. Ainsi le test du 0 ou du 1 baladeur se fait en 9 passes sur toute la mémoire et le damier bouclant en 2 passes. L'interface utilisateur de ce programme est effectuée avec 4 menus différents :
- le menu principal ou est demandé le type de carte RAM à tester.
- le menu secondaire ou est demandé le type de test à effectuer
- le menu «Test en cours» ou est affiché le compteur de cycles de test (en effet les tests bouclent indéfiniment sur eux mêmes).
- le menu de faute ou sont affichés le compteur de cycles, la page et l'adresse en faute, la valeur écrite (et attendue) avec la valeur lue.

La touche ESCape permet le retour au menu précédent, La touche Return ou Enter permet de continuer le test après un affichage de faute mémoire.
Le programme est constitué de plusieurs grands segments :
- L'initialisation et la saisie des informations nécessaires.
- Le test du 0 baladeur.
- Le test du 1 baladeur.
- Le test du damier bouclant.
- Le traitement de fin de cycle : affichage compteur, rebouclage sur le cycle suivant
- L'affichage des fautes mémoire
- Les routines d'affichage écran et d'attente du clavier

Chaque passe parmi 2 ou 9 ,selon le test choisi, est décomposé en deux parties :
Tout d'abord la carte est initialisée complètement avec le motif de test puis ce même motif est vérifié sur tous les octets, la vérification est figée sur l'adresse en faute qui est alors affichée sur l'écran. La frappe de la touche Return ou Enter permet de relancer le test sur l'octet suivant.
MISE EN OEUVRE :
Après avoir saisi le texte source à l'aide d'un éditeur comme EDLIN ou EDIT du MS-DOS ou encore EDITOR du DR-DOS, sauvegarder ce source dans un fichier nommé TEST.ASM.
Avec TASM (c) Borland, assemblez-le comme suit :
TASM /ML TEST... : Appel de l'Assembleur 80x86
TLINK /T TEST... : Appel de l'éditeur de liens
Avec MASM (c) MicroSoft, assembler le comme suit
MASM /ML TEST... : Appel de l'Assembleur 80x86
LINK TEST... : Appel de l'éditeur de liens
EXE2BIN TEST.EXE TEST.COM : Transformation en.COM
Le résultat de ces opérations est un fichier exécutable (TEST.COM) qui devra être chargé sur le disque virtuel C: du PORFOLIO avant d'être lancé.
Effectivement ce test est DESTRUCTIF (DESTRUCTIF) pour la carte à tester en A: ou B: , tous vos précieux fichiers présent sur ce support devront être sauvegardés avant de lancer ce test.

```
        PAGE    66,132
RC      EQU     0DH
ESCape  EQU     1BH
MENU    LABEL   BYTE
```

TEST MEMOIRE POUR PORTFOLIO... TEST MEMOIRE

```
        SEGMENT AT 0B000H
        ORG     0                       ;Adresse mémoire écran
SCRMEM          LABEL   WORD
SCREEN ENDS
CCM     LABEL   BYTE                    ;Zone CCM de 128K adressable
CCM1ST SEGMENT AT 0C000H
        ORG     0                       ;1er tranche de 64K
START1  LABEL   BYTE
        ORG     10                      ;Registre sélection de page de 128K
PageReg LABEL   BYTE
CCM1ST ENDS
CCM2ND          SEGMENT AT 0D000H
        ORG     0                       ;2em tranche de 64K
START2  LABEL   BYTE
CCM2ND          ENDS
SegSize EQU     10000H AND 0FFFFH       ;65536 octets par tranche de 64K
XSEG    SEGMENT
        ASSUME          CS:XSEG,DS:XSEG
        ORG     0100H
START:  MOV     AH,0                    ;Service Initialization
        INT     61H
        MOV     AX,(14 SHL 8)+0         ;GET screen mode
        INT     61H
        MOV     [SCRMOD],DL             ;Sauvegarde du mode
        MOV     AX,(14 SHL 8)+1         ;SET screen mode
        MOV     DL,00000001B            ;Format d'écran type PORTFOLIO
        INT     61H
        CLD                             ;Incrémentation
AFMEN1:         MOV     SI,OFFSET Menu1         ;DS:SI sur adresse 1er menu
        CALL    AFMENU
        MOV     AH,2                    ;Positionnement du curseur
        MOV     DX,(6 SHL 8)+34         ;Position du curseur
        INT     10H
        MOV     AX,(15 SHL 8)+0         ;GET cursor mode
        INT     61H
        MOV     [CURMOD],BL             ;Sauvegarde du mode
        MOV     AX,(15 SHL 8)+1         ;SET cursor mode
        MOV     BL,2                    ;Mode BLOCK pour ce dernier
        INT     61H
GETPAG:         MOV     AH,0                    ;Attente au clavier
        INT     16H
        CMP     AL,ESCape
        JE      RLABRT
        SUB     AL,'1'                  ;Test sur la réponse numérique
        JC      BADPAG
        CMP     AL,4
        JB      PAG_OK
BADPAG:         MOV     AX,(21 SHL 8)+1 ;Emission d'un beep
        INT     61H
        JMP     GETPAG
RLABRT:         JMP     ABORT
PAG_OK:         MOV     CL,AL           ;Calcul du Nombre de pages
        MOV     [NbPAGE],1
        SHL     [NbPAGE],CL
AFMEN2:         MOV     SI,OFFSET Menu2         ;DS:SI sur adresse 2em menu
        CALL    AFMENU
        MOV     AH,2                    ;Positionnement du curseur
        MOV     DX,(6 SHL 8)+34         ;Position du curseur
        INT     10H
GETTST: MOV     AH,0                    ;Attente au clavier
        INT     16H
        CMP     AL,ESCape
        JE      AFMEN1
        SUB     AL,'0'                  ;Test sur la réponse numérique
        JC      BADTST
        CMP     AL,3
        JBE     TST_OK
BADTST:         MOV     AX,(21 SHL 8)+1         ;Emission d'un beep
        INT     61H
        JMP     GETTST
TST_OK:MOV      [NoTEST],AL
        MOV     [COUNT],0               ;Compteur de passes à 0
        CALL    AFMEN3                  ;Affichage menu 3 avec compteur à 0
TSTCCM:         MOV     AX,(36 SHL 8)+1 ;SET ROM/CCM state
        MOV     DX,(1 SHL 8)+1          ;Validation permanente du CCM A:
        INT     61H
        MOV     [RSTATE],DX             ;Sauvegarde état précédent
        MOV     BL,[NoTEST]
        MOV     BH,0
        SHL     BX,1
        JMP     WORD PTR [OFFSET TABJMP+BX]
TABJMP:DW       OFFSET BALAD1           ;Test du 1 baladeur
        DW      OFFSET BALAD0           ;Test du 0 baladeur
        DW
                OFFSET BALAD1

                                        ;Enchainement des 2 tests
        DW      OFFSET DAMIER           ;Test du damier bouclant
BALAD1:         MOV     DL,11111111B    ;Pas de 0 baladeur
TSCCM1:         MOV     DH,0            ;Numéro de page à 0
        MOV     AL,DL                   ;Référence de départ
WR1PAG:         MOV     DI,CCM1ST
        MOV     ES,DI                   ;Adresse 1er tranche de 64K
        ASSUME          ES:CCM1ST
        MOV     DI,OFFSET START1
        MOV     ES:[PageReg],DH         ;Sélection de la page
WRAL10:         STOS    CCM
        CMP     AL,11111111B            ;CF à 0 si aucun bit à 0,sinon CF à 1
        RCL     AL,1                    ;Entrée du nouveau bit par la droite
        CMP     DI,OFFSET PageReg       ;Adresse du registre de page
        JNE     WRFT10
        INC     DI                      ;Saut de cette adresse
        JMP     WRAL10
WRFT10:         CMP     DI,SegSize      ;Test si fin de 1er tranche
        JNE     WRAL10
        MOV     DI,CCM2ND
        MOV     ES,DI                   ;Adresse 2em tranche de 64K
        ASSUME          ES:CCM2ND
        MOV     DI,OFFSET START2
WRAL11:         STOS    CCM
        CMP     AL,11111111B            ;CF à 0 si aucun bit à 0,sinon CF à 1
        RCL     AL,1                    ;Entrée du nouveau bit par la droite
        CMP     DI,SegSize              ;Test si fin de 2em tranche
        JNE     WRAL11
        INC     DH                      ;Changement de page
        CMP     DH,[NbPAGE]
        JB      WR1PAG
        MOV     DH,0                    ;Numéro de page à 0
        MOV     AL,DL                   ;Récupération de l'état de départ
CP1PAG: MOV     DI,CCM1ST
        MOV     ES,DI                   ;Adresse 1er tranche de 64K
        ASSUME          ES:CCM1ST
        MOV     DI,OFFSET START1
        MOV     ES:[PageReg],DH         ;Sélection de la page
CPAL10: SCAS    CCM
        JE      CPTB10
        CALL    DEFMEM                  ;Affichage du défaut
        JNC     CPTB10
        JMP     AFMEN2
CPTB10: CMP     AL,11111111B            ;CF à 0 si aucun bit à 0,sinon CF à 1
        RCL     AL,1                    ;Entrée du nouveau bit par la droite
        CMP     DI,OFFSET PageReg       ;Adresse du registre de page
        JNE     CPFT10
        INC     DI                      ;Saut de cette adresse
        JMP     CPAL10
CPFT10: CMP     DI,SegSize              ;Test si fin de 1er tranche
        JNE     CPAL10
        MOV     DI,CCM2ND
        MOV     ES,DI                   ;Adresse 2em tranche de 64K
        ASSUME          ES:CCM2ND
        MOV     DI,OFFSET START2
CPAL11: SCAS    CCM
        JE      CPTB11
        CALL    DEFMEM                  ;Affichage du défaut
        JNC     CPTB11
        JMP     AFMEN2
CPTB11: CMP     AL,11111111B            ;CF à 0 si aucun bit à 0,sinon CF à 1
        RCL     AL,1                    ;Entrée du nouveau bit par la droite
        CMP     DI,SegSize              ;Test si fin de 2em tranche
        JNE     CPAL11
        INC     DH                      ;Changement de page
        CMP     DH,[NbPAGE]
        JB      CP1PAG
        CALL    INCCNT                  ;Incrément et affichage du compteur
        CMP     DL,11111111B            ;CF à 0 si aucun bit à 0,sinon CF à 1
        RCL     DL,1                    ;Entrée du nouveau bit par la droite
        CMP     DL,11111111B            ;le 0 a-t-il été baladé sur tous les bits
        JNE     RTCCM1
        CMP     [NoTEST],2              ;Test si double test No 0 et 1
        JE      BALAD0                  ;Enchainement des 2 tests
        JMP     ENDTST
RTCCM1:         JMP     TSCCM1                  ;Relais avec saut NEAR
BALAD0:         MOV     DL,00000000B            ;Pas de 1 baladeur
TSCCM0:         MOV     DH,0            ;Numéro de page à 0
        MOV     AL,DL                   ;Référence de départ
WR0PAG:         MOV     DI,CCM1ST
        MOV     ES,DI                   ;Adresse 1er tranche de 64K
        ASSUME          ES:CCM1ST
        MOV     DI,OFFSET START1
```

TEST MEMOIRE POUR PORTFOLIO... TEST MEMOIRE

```
        JNE     WRSH00          ;CF toujours à 0, ZF à 1 si aucun bit à 1
        STC
WRSH00:         RCL     AL,1            ;Entrée du nouveau bit par la droite
        CMP     DI,OFFSET PageReg       ;Adresse du registre de page
        JNE     WRFT00
        INC     DI                      ;Saut de cette adresse
        JMP     WRAL00
WRFT00:         CMP     DI,SegSize      ;Test si fin de 1er tranche
        JNE     WRAL00
        MOV     DI,CCM2ND
        MOV     ES,DI                   ;Adresse 2em tranche de 64K
        ASSUME          ES:CCM2ND
        MOV     DI,OFFSET START2
WRAL01:         STOS    CCM
        CMP     AL,00000000B            ;Test si le bit est disparu
        JNE     WRSH01                  ;CF toujours à 0, ZF à 1 si aucun bit à 1
        STC
WRSH01:         RCL     AL,1            ;Entrée du nouveau bit par la droite
        CMP
                DI,SegSize              ;Test si fin de 2em tranche
        JNE     WRAL01
        INC     DH                      ;Changement de page
        CMP     DH,[NbPAGE]
        JB      WR0PAG
        MOV     DH,0                    ;Numéro de page à 0
        MOV     AL,DL                   ;Récupération de l'état de départ
CP0PAG: MOV     DI,CCM1ST
        MOV     ES,DI                   ;Adresse 1er tranche de 64K
        ASSUME          ES:CCM1ST
        MOV     DI,OFFSET START1
        MOV     ES:[PageReg],DH         ;Sélection de la page
CPAL00: SCAS    CCM
        JE      CPTB00
        CALL    DEFMEM                  ;Affichage du défaut
        JNC     CPTB00
        JMP     AFMEN2
CPTB00: CMP     AL,00000000B            ;Test si le bit est disparu
        JNE     CPSH00                  ;CF toujours à 0, ZF à 1 si aucun bit à 1
        STC
CPSH00: RCL     AL,1                    ;Entrée du nouveau bit par la droite
        CMP     DI,OFFSET PageReg       ;Adresse du registre de page
        JNE     CPFT00
        INC     DI                      ;Saut de cette adresse
        JMP     CPAL00
CPFT00: CMP     DI,SegSize              ;Test si fin de 1er tranche
        JNE     CPAL00
        MOV     DI,CCM2ND
        MOV     ES,DI                   ;Adresse 2em tranche de 64K
        ASSUME          ES:CCM2ND
        MOV     DI,OFFSET START2
CPAL01: SCAS    CCM
        JE      CPTB01
        CALL    DEFMEM                  ;Affichage du défaut
        JNC     CPTB01
        JMP     AFMEN2
CPTB01: CMP     AL,00000000B            ;Test si le bit a disparu
        JNE     CPSH01                  ;CF toujours à 0, ZF à 1 si aucun bit à 1
        STC
CPSH01: RCL     AL,1                    ;Entrée du nouveau bit par la droite
        CMP     DI,SegSize              ;Test si fin de 2ème tranche
        JNE     CPAL01
        INC     DH                      ;Changement de page
        CMP     DH,[NbPAGE]
        JB      CP0PAG
        CALL    INCCNT                  ;Incrément et affichage ducompteur
        CMP     DL,00000000B            ;Test si le bit est disparu
        JNE     SHLDL0                  ;CF toujours à 0, ZF à 1 si aucun bit à 1
        STC
SHLDL0:RCL      DL,1                    ;Entrée du nouveau bit par la droite
        CMP     DL,00000000B            ; le 1 a-t-il été baladé sur tous les bits
        JNE     RTCCM0
        JMP     ENDTST
RTCCM0:         JMP     TSCCM0          ;Relais avec saut NEAR
DAMIER:         MOV     DL,01010101B    ;Départ damier (55H/0AAH)
DAMTST:         MOV     DH,0            ;Numéro de page à 0
        MOV     AL,DL                   ;Référence de départ
WRPAG0:         MOV     DI,CCM1ST
        MOV     ES,DI                   ;Adresse 1er tranche de 64K
        ASSUME          ES:CCM1ST
        MOV     DI,OFFSET START1
        MOV     ES:[PageReg],DH         ;Sélection de la page
WRDAM0:         STOS    CCM
        CMP     DI,OFFSET PageReg       ;Adresse du registre de page
        JNE     WRINV0
        JMP     WRDAM0
WRINV0:         NOT     AL              ;Inversion du damier
        CMP     DI,SegSize              ;Test si fin de 1er tranche
        JNE     WRDAM0
        MOV     DI,CCM2ND
        MOV     ES,DI                   ;Adresse 2em tranche de 64K
        ASSUME          ES:CCM2ND
        MOV     DI,OFFSET START2
WRDAM1:         STOS    CCM
        NOT     AL                      ;Inversion du damier
        CMP     DI,SegSize              ;Test si fin de 2em tranche
        JNE     WRDAM1
        INC     DH                      ;Changement de page
        CMP     DH,[NbPAGE]
        JB      WRPAG0
        MOV     DH,0                    ;Numéro de page à 0
        MOV     AL,DL                   ;Récupération de l'état de départ
CPPAG0: MOV     DI,CCM1ST
        MOV     ES,DI                   ;Adresse 1er tranche de 64K
        ASSUME          ES:CCM1ST
        MOV     DI,OFFSET START1
        MOV     ES:[PageReg],DH         ;Sélection de la page
CPDAM0:         SCAS    CCM
        JE      CP_OK0
        CALL    DEFMEM                  ;Affichage du défaut
        JNC     CP_OK0
        JMP     AFMEN2
CP_OK0: CMP     DI,OFFSET PageReg       ;Adresse du registre de page
        JNE     CPINV0
        INC     DI                      ;Saut de cette adresse
        JMP     CPDAM0
CPINV0: NOT     AL
        CMP     DI,SegSize              ;Test si fin de 1er tranche
        JNE     CPDAM0
        MOV     DI,CCM2ND
        MOV     ES,DI                   ;Adresse 2em tranche de 64K
        ASSUME          ES:CCM2ND
        MOV     DI,OFFSET START2
CPDAM1:         SCAS    CCM
        JE      CP_OK1
        CALL    DEFMEM                  ;Affichage du défaut
        JNC     CP_OK1
        JMP     AFMEN2
CP_OK1: NOT     AL
        CMP     DI,SegSize              ;Test si fin de 2em tranche
        JNE     CPDAM1
        INC     DH                      ;Changement de page
        CMP     DH,[NbPAGE]
        JB      CPPAG0
        CALL    INCCNT                  ;Incrément et affichage du compteur
        NOT     DL
        CMP     DL,01010101B            ;Le damier a t-il été testé sur chaque octet ?
        JE      ENDTST
        JMP     DAMTST
ENDTST:         MOV     AX,(36 SHL 8)+1 ;Fonction SET ROM/CCM state
        MOV     DX,[RSTATE]             ;Récupération état précédent
        INT     61H
        MOV     AX,(21 SHL 8)+1         ;Emission d'un beep à chaque passe
        INT     61H
        MOV     AH,1                    ;Vérification du clavier
        INT     16H
        JZ      NO_BRK
        MOV     AH,0                    ;Prise en compte du caractère
        INT     16H
        CMP     AL,ESCape
        JE      RLAFM2
NO_BRK:         JMP     TSTCCM
RLAFM2:         JMP     AFMEN2
INCCNT: INC     [COUNT]         ;Incrément du nombre de passes
AFMEN3:         PUSH    AX      ;Affichage du menu 3 avec le compteur
        PUSH    DX
        PUSH    DI
        PUSH    ES
        MOV     AL,BYTE PTR [COUNT+1]   ;Affichage du compteur de passes
        CALL    ASC_L
        MOV     CS:[Menu3+206],AL
        MOV     AL,BYTE PTR [COUNT]     ;Affichage octet inférieur du compteur
        CALL    ASC_H
        MOV     CS:[Menu3+207],AL
        MOV     AL,BYTE PTR [COUNT]
        CALL    ASC_L
        MOV     CS:[Menu3+208],AL
        MOV     SI,OFFSET Menu3         ;DS:SI sur adresse 3em menu
        CALL    AFMENU
        MOV     AH,2                    ;Positionnement du curseur
        MOV     DX,(5 SHL 8)+10         ;Position du curseur
        INT     10H
```

TEST MEMOIRE POUR PORTFOLIO... TEST MEMOIRE

```
        JMP     RETOUR
DEFMEM:         PUSH    AX                      ;Affichage d'un défaut mémoire
        PUSH    DX
        PUSH    DI
        PUSH    ES
        DEC     DI
        MOV     AH,AL                           ;Sauvegarde référence écrite
        PUSH    AX
        MOV     AL,BYTE PTR [COUNT+1]           ;Affichage du compteur de passes
        CALL    ASC_L
        MOV     CS:[Menu4+205],AL
        MOV     AL,BYTE PTR [COUNT]             ;Affichage octet inférieur du compteur
        CALL    ASC_H
        MOV     CS:[Menu4+206],AL
        MOV     AL,BYTE PTR [COUNT]
        CALL    ASC_L
        MOV     CS:[Menu4+207],AL
        MOV     AL,DH                           ;Affichage du No de page
        CALL    ASC_L
        MOV     CS:[Menu4+213],AL
        MOV     AX,ES
        MOV     AL,AH
        AND     AL,00010000B                    ;No de tranche de 64K
        CALL    ASC_H
        MOV     CS:[Menu4+218],AL
        MOV     AX,DI                           ;Affichage octet supérieur de l'adresse
        MOV     AL,AH
        CALL    ASC_H
        MOV     CS:[Menu4+219],AL
        MOV     AL,AH
        CALL    ASC_L
        MOV     CS:[Menu4+220],AL
        MOV     AX,DI                           ;Affichage octet inférieur de l'adresse
        CALL    ASC_H
        MOV     CS:[Menu4+221],AL
        MOV     AX,DI
        CALL    ASC_L
        MOV     CS:[Menu4+222],AL
        POP     AX                              ;Affichage de la donnée écrite
        CALL    ASC_H
        MOV     CS:[Menu4+229],AL
        MOV     AL,AH
        CALL    ASC_L
        MOV     CS:[Menu4+230],AL
        MOV     AL,ES:[DI]                      ;Affichage de la donnée lue
        CALL    ASC_H
        MOV     CS:[Menu4+235],AL
        MOV     AL,ES:[DI]
        CALL    ASC_L
        MOV     CS:[Menu4+236],AL
        MOV     SI,OFFSET Menu4                 ;DS:SI sur adresse 4em menu
        CALL    AFMENU
        MOV     AH,2                            ;Positionnement du curseur
        MOV     DX,(5 SHL 8)+9                  ;Position du curseur
        INT     10H
WAITK:  MOV     AH,0                            ;Attente au clavier
        INT     16H
        CMP     AL,ESCape
        JNE     TESTK
        MOV     AX,(36 SHL 8)+1                 ;SET ROM/CCM state
        MOV     DX,[RSTATE]                     ;Récupération état précédent
        INT     61H
        STC
        JMP     RETOUR
TESTK:  CMP     AL,RC
        JNE     BEEP
        MOV     SI,OFFSET Menu3                 ;DS:SI sur adresse 1er menu
        CALL    AFMENU
        MOV     AH,2                            ;Positionnement du curseur
        MOV     DX,(5 SHL 8)+32                 ;Position du curseur
        INT     10H
        CLC
        JMP     RETOUR
BEEP:   MOV     AX,(21 SHL 8)+1                 ;Emission d'un beep
        INT     61H
        JMP     WAITK
RETOUR:         POP     ES
        POP     DI
        POP     DX
        POP     AX
        RET
ASC_H:  SHR     AL,1                            ;Isolement quartet de gauche
        SHR     AL,1
        SHR     AL,1
        SHR     AL,1
ASC_L:  AND     AL,0FH                          ;Conversion HEXA/ASCII d'un quartet
        ADD     AL,90H
        DAA
        ADC     AL,40H
        DAA
        RET
ABORT:  MOV     AX,(0 SHL 8)+7                  ;Mode vidéo du MDA pour effacer l'écran
        INT     10H
        MOV     AX,(14 SHL 8)+1                 ;SET screen mode
        MOV     DL,[SCRMOD]                     ;Récupération format d'écran
        INT     61H
        MOV     AX,(15 SHL 8)+1                 ;SET cursor mode
        MOV     BL,[CURMOD]                     ;Récupération du mode
        INT     61H
        MOV     AX,(76 SHL 8)+0                 ;Retour DOS avec ERRORLEVEL=0
        INT     21H
        JMP     ABORT
AFMENU:         MOV     DI,SCREEN
        MOV     ES,DI
        ASSUME          ES:SCREEN
        MOV     DI,OFFSET SCRMEM                ;ES:DI en debut d'ecran virtuel
        MOV     AH,00000111B                    ;Attribut normal video
        MOV     CH,8                            ;Nombre de lignes sur l'écran
AFMN00:         MOV     CL,40                   ;Nombre de caractères par ligne
AFMN01:         LODS    MENU                    ;Lecture de chaque caractère dans AL
        STOS    SCRMEM                          ;Ecriture du caractère et de son attribut
        DEC     CL                              ;Affichage d'une ligne
        JNZ     AFMN01
        ADD     DL,(40*2)                       ;Saut des 40 caractères non affichables
        DEC     CH
        JNZ     AFMN00
        MOV     AH,18                           ;Screen Refresh
        INT     61H
        RET
SCRMOD          LABEL   BYTE                    ;Sauvegarde du mode écran
        DB      ?
CURMOD          LABEL   BYTE                    ;Sauvegarde du type de curseur
        DB      ?
NbPAGE LABEL    BYTE                            ;Nombre de pages de 128K
        DB      ?
NoTEST LABEL    BYTE                            ;Numéro de test à effectuer
        DB      ?
COUNT  LABEL    WORD                            ;Compteur de passes
        DB      ?
RSTATE LABEL    WORD                            ;Etat espace mémoire ROM/CCM
        DW      ?
Menu1   LABEL   BYTE
        DB      `+————————————————————————————+`
        DB      `+—Á TEST carte mémoire multipages â—+`
        DB      `¦ +————————————————————————+ ¦`
        DB      `¦ (1) : carte 128K ¦ ESCape : abandon ¦`
        DB      `¦ (2) : carte 256K +————————————¦`
        DB      `¦ (3) : carte 512K ¦  Entrez votre  ¦`
        DB      `¦ (4) : carte  1M ¦   choix : _   ¦`
        DB      `+————————————————————————————+`
Menu2   LABEL   BYTE
        DB      `+————————————————————————————+`
        DB      `+—Á TEST carte mémoire multipages â—+`
        DB      `¦ +————————————————————————+ ¦`
        DB      `¦ (0) : 0 Baladeur ¦ ESCape : 1er MENU ¦`
        DB      `¦ (1) : 1 Baladeur +————————————¦`
        DB      `¦ (2) : Tests 0 & 1 ¦  Entrez votre  ¦`
        DB      `¦ (3) : Damier     ¦   choix : _   ¦`
        DB      `+————————————————————————————+`
Menu3   LABEL   BYTE
        DB      `+————————————————————————————+`
        DB      `+—Á TEST  carte mémoire multipages â  —+`
        DB      `¦ +————————————————————————+ ¦`
        DB      `¦      ESCape : retour 2em MENU      ¦`
        DB      `¦ +————————————————————————+ ¦`
        DB      `¦ ¦ C:000 ¦ TEST mémoire en cours... ¦ ¦`
        DB      `¦ +————————————————————————+ ¦`
        DB      `+————————————————————————————+`
Menu4   LABEL   BYTE
        DB      `+————————————————————————————+`
        DB      `+—Á FAUTE carte mémoire multipages â—+`
        DB      `¦ +————————————————————————+ ¦`
        DB      `¦ ESCape : MENU  ou ENTER : suite test ¦`
        DB      `¦+————————————————————————————+¦`
        DB      `¦¦ C:___ ¦ P:_ AD:_____ ¦ WR:_  RD:__ ¦¦`
        DB      `¦+————————————————————————————+¦`
        DB      `+————————————————————————————+`
XSEG    ENDS
        END     START
```

AFFICHAGE GRAPHIQUE SUR ATARI PORTFOLIO

par Thierry BODIN

Ceux d'entre-vous qui ont voulu programmer sur leur Portfolio avec PBASIC, TURBO-PASCAL ou autre se sont vite rendu compte d'une chose: ce n'est pas très rapide. C'est pourquoi nous allons étudier quelques routines en Assembleur.

Mais pas n'importe lesquelles ! Des routines pour gérer le mode graphique : écriture ou lecture d'un pixel, copie de sprites, traceur de lignes, gestion du contraste pour afficher/faire disparaître une image progressivement (en fondu), affichage d'une image (possibilité d'afficher jusqu'à environ 50 images par secondes),... mais aussi des routines de gestion du son, du curseur, etc...
Bref, une mine d'or pour les programmeurs !
Ainsi, vous serez paré pour nous faire un jeu graphique, une démo ou une application graphique car notre Portfolio en manque cruellement. Vous remarquerez sans doute que certaines suites d'instructions se répètent pour avoir un gain de vitesse non négligeable sur le 8088.
Pour une meilleure compréhension, des routines sont utilisées dans l'exemple 1 qui anime un nombre défini de balles qui rebondissent sur l'écran. Dans l'exemple 2, ce sont des étoiles qui vont de droite à gauche.
Voilà, tout est dit. Ah non, encore une chose: si vous avez un problème pour utiliser ces routines, contactez-moi sur le 3615 RTEL, Boîte Aux Lettres (BAL) SCARFY. N'oubliez pas de valider votre pseudo en créant une BAL (en tapant *BAL et ENVOI) pour que je puisse vous répondre. Il existe aussi un club PORTFOLIO. Pour y accéder, tapez *LCB et ENVOI puis PORTFOLIO, le code est PORTFO. Autrement, si vous voulez recevoir la version de démonstration de mon casse-briques ou des informations plus techniques sur les ports, envoyez-moi une disquette+ enveloppe timbrée (pour la démo) ou écrivez-moi à: Thierry BODIN, «Launay-Hyon», 35230 Noyal/Seiche.
A bientôt !

```
*****************************************************
                    EXEMPLE N°1
          Routines graphiques pour Atari Portfolio
                  par Thierry BODIN
               Fichier: SCARFY1.ASM
        Pour assembler cet exemple, il faut utiliser
    Turbo Assembler de Borland (TASM) comme ceci:
                 TASM SCARFY1 /m4
                 TLINK SCARFY1 /t
      Tapez ensuite SCARFY1 au prompt du DOS
          Les «;» indiquent des commentaires
*****************************************************

JUMPS
    CODE   Segment
    Assume cs:CODE
    Org    100h          ; Nous allons compiler un
                         ; fichier .COM

NBBALLE equ    150       ; Nombre de balles à afficher

Début:
    push   cs
    pop    ds            ; DS=CS
    push   cs
    pop    es            ; ES=CS
    mov    ax,5          ; Fct 00h,mode 05h de l'Int 10h
    int    10h           ; Passage en mode graphique (240*64)
    mov    dx,8060h
    in     al,dx
    mov    ds:[oldcont],al ; Sauve le contraste de l'utilisateur
    call   GenerPosBalles  ; Routine pour tirer aléatoirement
                           ; la position de chaque balle
    mov    ax,0B000h       ; Segment vidéo
    mov    es,ax           ; Il est important que ES soit égal à B000h
                           ; avant d'appeler une des routines
recom:
    mov    bp,NBBALLE
    mov    si,offset Posballe
    xor    ah,ah
    mov    bh,ah
    cli                    ; Evite les parasites du rafraîchissement et
                           ; autres programmes externes
repu:
    lodsb                  ; Charge coordonnée Y
    mov    bl,al
    lodsb                  ; Charge coordonnée X
    push   bx              ; On transmet les coordonnées
    push   ax              ; par la pile
    call   AffBalle        ; Copie le sprite de la balle dans le segment vidéo
    pop    ax
    pop    bx
    inc    si              ; On ne lit pas encore les informations
    inc    si              ; sur le sens de déplacement de la balle
    dec    bp              ; Toutes les balles sont copiées ?
    jnz    repu            ; NON-> on recommence
    call   putimage        ; Envoi le segment vidéo au contrôleur écran
    sti
    mov    bp,NBBALLE
    mov    si,offset posballe : On calcule les nouvelles coordonnées
depbl:                     ; des balles
    lodsb                  ; Charge Y
    add    al,ds:[si+1]    ; Ajoute le sens de déplacement (-2,-1,1 ou 2)
    cmp    al,59           ; La balle est en bas de l'écran ?
    jae    chgdepy         ; OUI -> on change le sens de déplacement
    cmp    al,1            ; La balle est en haut de l'écran ?
    ja     paschgy         ; OUI -> on change le sens de déplacement
chgdepy:
    neg    byte ptr ds:[si+1] ;Inverse le sens: 1 devient -1,
```

```
                    ; -2 devient 2,...
paschgy:
      mov    ds:[si-1],al    ; Sauve la nouvelle coordonnée Y
     lodsb                  ; Charge X
      add    al,ds:[si+1]    ; Ajoute le sens de déplacement (-2,-1,1 ou 2)
      cmp    al,235          ; La balle est à droite de l'écran ?
      jae    chgdepx         ; OUI -> on change le sens de déplacement
      cmp    al,1            ; La balle est à gauche de l'écran ?
      ja     paschgX         ; OUI -> on change le sens de déplacement
chgdepx:
      neg    byte ptr ds:[si+1] ;Inverse le sens: 1 devient -1,
                    ; -2 devient 2,...
paschgX:
      mov    ds:[si-1],al    ; Sauve la nouvelle coordonnée X
      inc    si              ; On passe à la
      inc    si              ; balle suivante
      dec    bp              ; Il reste des balles à déplacer ?
      jnz    depbl           ; OUI -> On recommence
      mov    cx,8000         ; Pause pour éviter la rémanence
      loop   $
      mov    ah,1            ; Test si une touche a été appuyé
      int    16h
      jnz    norecom         ; OUI -> on sort

      mov    bp,NBBALLE
      mov    si,offset posballe
      xor    di,di
      mov    cx,1920/2
      xor    ax,ax
      rep    stosw           ; Efface le segment vidéo
      jmp    repu
norecom:
      xor    ah,ah
      int    16h             ; Vide la touche du tampon-clavier
      call   PalDown         ; Efface en fondu grâce au réglage du contraste
      mov    ax,7            ; Fct 00h, mode 07h de l'Int 10h
      int    10h             ; Passe en mode texte
      mov    ah,9            ; Fct 09h, affichage d'une chaîne
      mov    dx,offset msg   ; Affiche le (c)
      int    21h
      call   PalUp           ; Remet le contraste d'origine en fondu
      mov    ax,4C00h        ; Fct 4Ch, retour au DOS. Le 00h
      int    21h             ; signifie qu'il n'y a pas eu d'erreur
;*****************************************************;
;* Copie le sprite d'une balle dans le segment vidéo *;
;*****************************************************;
; Syntaxe d'appel:
; Push  Y
; Push  X
; Call  AffBalle
; add   sp,4
AffBalle proc near
      push   bp
      mov    bp,sp
      cli                    ; Empêche le rafraichissement d'écran
      push   si
      mov    di,4            ; Le sprite fait 4 lignes
      mov    si,offset babal ; Pointe sur le sprite
reaffballe:
      mov    ax,ss:[bp+6]    ; AX= N° de ligne (0 à 63)
      mov    bx,ax
      shl    bx,1            ; Multiplie BX par 2
      shl    ax,1
      shl    ax,1
      shl    ax,1
      shl    ax,1
      shl    ax,1            ; Multiplie AX par 32
      sub    ax,bx           ; 32-2=30 donc =AX*30
                    ; On multiplie Y par 30 car l'écran fait 30
                    ; caractères de 8 bits (soit 30*8 pixels=240)
                    ; par ligne
      mov    dx,ss:[bp+4]    ; DX= N° de colonne (0 à 239)
      mov    bl,dl
      and    bl,111b         ; BL= N° de bit d'origine pour copier le sprite
      shr    dx,1
      shr    dx,1
      shr    dx,1            ; Divise DX par 8 pour avoir le n° d'octet
                    ; à lire
      add    dx,ax           ; (de 0 à 29) de la ligne
      mov    cx,dx
      xchg   bx,cx           ; BX= Offset vidéo
                    ; CL= N° de bit d'origine
      push   bx
      mov    bx,es:[bx]      ; Lit dans le segment vidéo ce qui est déjà
                    ; affiché pour faire un masque
      ror    bx,cl           ; Positionne sur le 1er bit à modifier
      and    bl,11110000b    ; La balle fait 4 bits de long donc on libère
                    ; la place pour la copier (Elle sera copiée sur
                    ; les 0000)
      lodsb                  ; Charge une ligne de la balle
      add    bl,al           ; L'ajoute
      rol    bx,cl           ; Replace les bits
      mov    ax,bx
      pop    bx
      mov    es:[bx],ax      ; Rajoute dans le segment
      inc    word ptr ss:[bp+6] ; Ligne suivante
      dec    di              ; Les 4 lignes sont copiées ?
      jnz    reaffballe      ; NON -> on recommence
      pop    si
      sti                    ; Autorise le rafraîchissement
      mov    bp,sp
      pop    bp
      retn                   ; On sort
AffBalle endp
;*****************************************;
;* Recopie le segment vidéo à l'écran *;
;*****************************************;
; NOTE IMPORTANTE: Ce que vous copiez dans le segment vidéo ne sera pas
; affiché immédiatement car le rafraîchissement n'a lieu que toutes les
; 128 secondes ou toutes les secondes selon votre configuration.
; Cette routine  passe outre et copie le contenu du segment sur l'écran
; via des OUTs. Il est obligatoire d'écrire les  pixels dans le segment
; vidéo avant de les envoyer sinon, en cas de rafraîchissement du BIOS,
; votre image sera remplacée par le contenu du segment.
; Syntaxe d'appel:
; Call  PutImage
PutImage proc near
      push   ds
      push   es
      pop    ds
      mov    cx,30*64        ; 1920 octets à envoyer
      xor    si,si
      mov    dx,8011h        ; Fct 0Ah du port 8011h
      mov    ax,000Ah        ; Envoyer l'octet de poids faible de l'adresse
      out    dx,al
      mov    al,ah
      dec    dx
      out    dx,al           ; Port 8010h, Envoi de l'octet de poids faible
      inc    dx
      mov    al,0Bh          ; Fct 0Bh du port 8011h
      out    dx,al           ; Envoyer l'octet de poids fort de l'adresse
      mov    al,ah
      dec    dx
      out    dx,al           ; Port 8010h, Envoi de l'octet de poids fort
      inc    dx
      mov    al,0Ch          ; Fct 0Ch du port 8011h
      out    dx,al           ; Envoyer le caractère à afficher
      dec    dx
pset_3:
```

```
5 n=1: kill fitemp$: pgsave fitemp$ return

'**** Sortie de programme
    lodsb           ; Charge le caractère
    out   dx,al     ; et l'envoie dans le port 8010h
    nop             ; Ces 4 NOPs font perdre du temps à la
    nop             ; machine pour que le contrôleur vidéo
    nop             ; ait le temps de suivre, sinon il affiche
    nop             ; n'importe quoi.
    loop  pset_3    ; Envoie tous les caractères
    pop   ds
    retn
PutImage endp
;*************************************;
;* Permet de faire un tirage aléatoire *;
;*************************************;
; Renvoie dans AL un nombre compris entre
; 0 et nn, nn étant la puissance de 2 - 1
; du masque contenu dans AH.
; Pour obtenir un nombre entre 0 et 65535
; il faut lire la variable MALEAT.
; Je tiens à remercier Rod et Thierry
; (Sysop & CoSysop de RTEL) qui ont créé
; cette routine.
; Syntaxe d'appel:
; Call Random
Random proc near
    push  cx
    mov   ch,8          ; Valeur de la boucle...
Baleat:
    mov   al,byte ptr ds:[Maleat] ; Mémoire aléatoire
    sal   al,1
    xor   al,byte ptr ds:[Maleat+1]
    sal   al,1
    sal   al,1
    rcl   byte ptr ds:[Maleat],1
    rcl   byte ptr ds:[Maleat+1],1
    inc   word ptr ds:[Maleat]
    dec   ch
    jne   Baleat
    mov   al,byte ptr ds:[Maleat]
    and   al,ah
    xor   ah,ah
    pop   cx
    retn
Random endp
Maleat dw   0ACDCh      ;Nombre aléatoire.
;*****************************************************************;
;* Génère la position et le sens de déplacement de chaque balle *;
;*****************************************************************;
; Syntaxe d'appel:
; Call GenerPosBalles
GenerPosBalles proc near
    mov   di,offset posballe
    mov   bp,NBBALLE
posbalrnd:
    mov   ah,63         ; Tirage de Y
    call  Random
    add   al,2
    cmp   al,58
    ja    posbalrnd
    stosb
    mov   ah,0FFh
realeat1:
    call  Random        ; Tirage de X
    add   al,2
    cmp   al,234
    ja    realeat1
    stosb
    mov   cx,2
rnddep:
    mov   ah,11b        ; Tirage des 2 sens de déplacement
    call  Random
    cmp   al,1
    jbe   depneg
    dec   al
    jmp   dep1good
depneg:
    sub   al,2
dep1good:
    stosb
    loop  rnddep
    dec   bp
    jnz   posbalrnd     ; Recommence pour toutes les balles
    retn
GenerPosBalles endp
;*************************************;
;* Efface en fondu grâce au contraste *;
;*************************************;
; Syntaxe d'appel:
; Call PalDown
PalDown proc near
    mov   dx,8060h      ; Port 8060h= Lire ou écrire la valeur
                        ; de réglage du contraste
    mov   al,ds:[oldcont]
    cli
reout:
    out   dx,al
    mov   cx,5500       ; On fait une pause pour voir l'effet
    loop  $
    dec   al            ; Tant que l'écran n'est pas clair on
    jnz   reout         ; recommence
    out   dx,al
    sti
    retn
PalDown endp
;****************************;
;* Affiche l'écran en fondu *;
;****************************;
; Syntaxe d'appel:
; Call Palup
PalUp proc near
    mov   dx,8060h      ; Port 8060h= Lire ou écrire la valeur
                        ; de réglage du contraste
    xor   al,al         ; On commence à 0
    cli
reout_2:
    out   dx,al
    mov   cx,5500       ; On fait une pause pour voir l'effet
    loop  $
    inc   al
    cmp   al,ds:[oldcont] ; Tant que l'écran n'est pas affiché,
    jbe   reout_2       ; on recommence
    sti
    retn
PalUp endp
;*****************************************************;
;*                Les données                        *;
;*****************************************************;
oldcont db  0
msg     db  '(c) 1994 Scarfy / Thierry BODIN',0Dh,0Ah,'$'

babal   db  0110b       ; Voici notre balle
        db  1011b
        db  1111b
        db  0110b
Posballe    db NBBALLE*4 dup (0)  ;1er octet = Coord. Y
                        ;2ème octet= Coord. X
                        ;3ème octet= Sens de Y
                        ;4ème octet= Sens de X
CODE Ends
End Début               ; Fin du programme
```

Fin de la 1ère partie.
Le prochain numéro, nous verrons le deuxième exemple.

L'EXTERNAL SUR HP48

par Mickaël Saillot

Pour certain, l'assembleur est trop compliqué malgré sa rapidité d'exécution imbattable. La seule solution étant donc de programmer avec le langage usuel de votre HP, mais vous vous apercevrez programme après programme que ce langage est lent, gourmand en mémoire et qu'il vous faudrait plus de commandes pour réaliser des programmes géniaux. La solution consiste donc à programmer en RPL - système, c'est à dire en External. Cet article va essayer de vous éclaircir sur la programmation en External en vous démontrant que les ingénieurs de chez HP on vraiment pensé à tout pour cette machine.

I) Qu'est-ce que l'External ?

Le langage External n'est en fait qu'une extension du RPL, seul langage auquel Hewlett Packard vous permet d'accéder. Toutes les commandes auxquelles vous avez accès (telles que DROP, SWAP...) sont programmées en External. En effet, celles-ci font appel à différentes adresses de la ROM du HP 48 vérifiant les arguments nécessaires sur la pile pour éviter le plantage du HP 48. Ce langage consiste à appeler une adresse de la ROM du HP 48 qui exécutera une routine interne ou déposera sur la pile un objet quelconque (il est important de savoir qu'un objet désigne les éléments de base d'information utilisés par le HP 48. Notons comme principal objet le programme, la chaîne de caractères, le réel, la liste, le graphique, etc.). N'ayant pas une multitude de tests et de contrôles à effectuer, la programmation en External permet de faire des programmes beaucoup plus rapides (de 2 à 7 fois; surtout pour les routines internes ayant rapport avec le graphisme). Et, de plus, le fait d'appeler une adresse dans la ROM du HP 48 ne nécessite que 5 quartets, c'est à dire 2,5 octets de mémoire. Vous devez sans doute pensez que l'on pourrait utiliser l'instruction SYSEVAL, ceci est vrai pour certaines routines, mais cette instruction vérifie la présence d'un binaire sur le niveau 1 de la pile donc il en résulte une perte de temps.

II) Comment programmer en External ?

Si vous voulez programmer en External il vous faudra connaître le maximum d'adresses possible sachant qu'appeler une adresse débouche sur l'évaluation d'un programme en général. (A ce propos, notez les excellents ouvrages de M. Jean Michel FERRARD : «les secrets de la HP 48» aux éditions D3I). La connaissance du microprocesseur de votre HP 48 n'est pas vraiment importante, au contraire de l'assembleur. Sachez que l'on peut appeler une adresse si et seulement si celle ci est inférieure à #70000h, c'est à dire #458752 en décimal. Vous allez trouver ci dessous différentes adresses couvrant les principaux domaines du RPL système. Ensuite on expliquera comment utiliser ces adresses afin d'obtenir un programme exécutable.

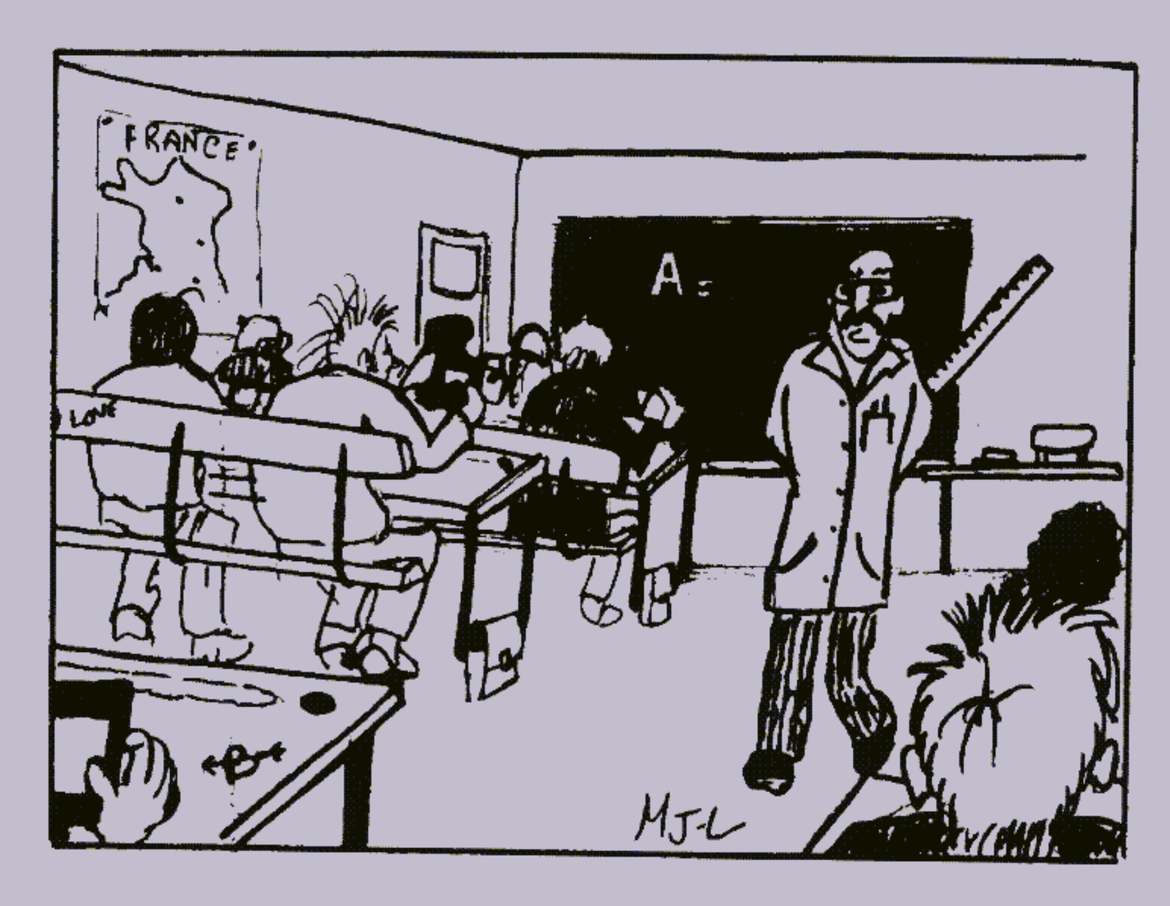
A) Les Booléens

Un booléen est un drapeau pouvant prendre la valeur vrai (true) ou la valeur faux (false). Celui ci est déposé sur le niveau 1 de la pile après un test. L'adresse du booléen true est #3A81h et celle du booléen false est #3AC0h. Si on actionne la touche EVAL sur un booléen présent au niveau 1, il ne se passe rien; ils sont donc inévaluables.

Quelques adresses :

#3AF2h : cette adresse permet d'effectuer une inversion du booléen présent au niveau 1 de la pile. (Ex : false devient true).

#3B75h : cette adresse effectue un test sur les deux booléens présents aux niveaux 1 et 2. Elle teste si au moins un des deux booléen est true.



#3ADAh : cette adresse teste si les deux booléens aux niveaux 1 et 2 sont différents.

#3B46h : cette adresse teste si les deux booléens présents aux niveaux 1 et 2 sont des booléens true.

B) Les entiers système

Un entier système est pour simplifier un chiffre pouvant prendre les valeurs de 0 à #FFFFFh (1048575 en décimal), qui est directement compréhensible par le microprocesseur de votre HP 48. Il est présenté sous la forme < XX...z > où «z» correspond à la base courante (d pour décimal, h pour hexadécimal). Il est impossible de le taper directement en ligne de commande. Un grand nombre de points d'entrés de votre HP 48 nécessitent des entiers système comme arguments.

Quelques adresses :

#3DBCh : additionne 2 entiers système.

#3DE0h : soustrait 2 entiers système.

#3EC2h : multiplie 2 entiers système.

#3DEFh : incrémente un entier système.

#3E0Eh : decrémente un entier système.

#3CA6h : teste si l'entier système présent sur le niveau 1 est égal à 0, la réponse est un booléen.

#3CC7h : teste si l'entier système présent sur le niveau 1 est différent de 0, la réponse est un booléen.

#3D19h : teste si les 2 entiers système présents sur les niveaux 1 et 2 sont égaux, la réponse est un booléen.

#3D4Eh : teste si les 2 entiers système présents sur les niveaux 1 et 2 sont différents, la réponse est un booléen.

#3CE4h : teste si l'entier système présent au niveau 1 est supérieur à celui du niveau 2, la réponse est un booléen.

#3D83h : teste si l'entier système présent au niveau 1 est infé-

rieur à celui du niveau 2, la réponse est un booléen.
#6264Eh : multiplie un entier système par 10.
#622B6h : teste si l'entier système présent au niveau 1 est égal à 1, la réponse est un booléen.
#636F0h : teste si l'entier système présent au niveau 1 est supérieur a 1 , la réponse est un booléen.

C) Opérations sur la pile

Il s'agit des fonctions les plus utilisées dans un programme permettant d'agir sur les objets présents sur la pile. En général les adresses font appel à des routines internes programmées en langage machine , il en résulte un gain de vitesse.
Quelques adresses:
#3244h : forme interne de DROP; élimine l'objet du niveau 1.
#3188h : forme interne de DUP; duplique l'objet du niveau 1.
#3223h : forme interne de SWAP; échange les niveaux 1 et 2.
#3258h : forme interne de DROP2; élimine les objets des niveaux 1 et 2 .
#3295h : forme interne de ROT; l'objet du niveau 3 apparaît au niveau 1.
#611FEh : effectue un 3 PICK; l'objet du niveau 3 est copié au niveau 1.
#60FBBh : effectue un 4 ROLL; l'objet du niveau 4 apparaît au niveau 1.
#60FACh : effectue un 3 ROLLD; l'objet du niveau 1 est placé au niveau 3.
#60F9Bh : effectue un DROP sur le niveau 2.
#60F21h : effectue un DROP sur le niveau 3 .
#62726h : effectue un DROP sur les niveaux 1 et 3.
#60EE7h : échange les niveaux 2 et 3.
#6270Ch : effectue DROP et ensuite SWAP.
#62747h : effectue SWAP et ensuite DUP.
#62830h : effectue un DROP sur le niveau 2 et ensuite DUP.
#62FB1h : effectue un DUP et ensuite ROT.
#62F89h : effectue un DROP sur le niveau représenté par un entier système au niveau 1.
#62D31h : effectue OVER et ensuite SWAP.
#6386Ch : effectue SWAP et ensuite DUP2.
#62CA5h : effectue ROT et ensuite OVER.
#60F33h : échange les niveaux 1 et 3.
#31D9h : effectue la fonction DUPN avec l'entier système présent au niveau 1.
#32E2h : effectue une copie du niveau donné par l'entier système présent au niveau 1.

D) Tests et exploitations des résultats

Certaines adresses de la ROM du HP 48 effectuent des tests. (Voir comme exemple les entiers système). Vous pouvez exploiter le résultat du test et diriger la suite de votre programme suivant que le booléen soit false ou true.

a) Structure IF THEN END

On a d'abord un test renvoyant un booléen. L'adresse #619BCh exécute l'objet suivant du programme si et seulement si le booléen est true. Le programme continue ensuite.

b) Structure IF THEN ELSE END

On a d'abord un test renvoyant un booléen. L'adresse #61AD8h exécute l'objet 1 si le booléen est true sinon elle exécute l'objet 2. Le programme continue ensuite.

c) Structure ELSE END

On a d'abord un test renvoyant un booléen. L'adresse #712Ah exécute l'objet suivant si et seulement si le booléen est false. Le programme continue ensuite.

E) Les boucles

a) DO prog UNTIL END

Le prog peut être constitué d'un ou plusieurs objets mais doit renvoyer un booléen nécessaire pour UNTIL END.
L'adresse de DO est #71A2h et celle du UNTIL END est #71C8h. Si le booléen est égal à true alors on sort de la boucle et on continue la suite du programme sinon on retourne à DO. On peut remplacer l'adresse #71C8h par #71ABh. Celle ci ne nécessite pas de booléen mais la boucle est exécutée sans fin.

b) WHILE REPEAT END

#712Ah : tant que ... (même adresse que le DO).
Test renvoyant un booléen.
#71EEh : exécute prog.
#71E5h : jusqu'à ce que le booléen du test soit false.

c) FOR I = 1 TO x prog NEXT

Les boucles non conditionnelles utilisent les entiers système comme arguments. L'adresse #7221h permet de ressortir sous forme d'entier système, l'indice de la boucle.

niveau 2 niveau 1	<n>	<n>	<n>	<n> <m>
For =	#73CEh 1 to n+1	# 73C3h 0 to n+1	#73DBh 1 to n	#73F7h m to n+1
niveau 1				<s>
	# 7334h NEXT	# 7334h NEXT	# 7334h NEXT	#73A5h <s>

F) Le graphisme.

La ROM du HP 48 possède énormément de points d'entrés ayant rapport avec le graphisme. Vous avez deux solutions soit utiliser l'environnement PICT (celui ou se dessine les courbes) ou utiliser le grob de la pile (celui que vous utilisez pour faire 2 + 2 !!). Je vous conseille d'utiliser le grob de la pile car les instructions sont beaucoup plus rapide et de plus vous n'effacez pas l'environnement PICT.
Quelques adresses:
#5046Ah : efface l'écran en entier.
#4E2CFh : efface la ligne des menus, vous permet d'utiliser le grob de la pile en entier c'est à dire 131*64.
#128B0h : applique le grob du niveau 1 aux coordonnées X et Y données par les entiers système présents aux niveaux 2 et 3.
#11D00h : effectue 1->GROB sur une chaîne de caractères.
#122FFh : inversion des points du grob présent au niveau 1 (un point blanc devient noir...).
#50758h : trace une ligne dans PICT , nécessite 4 entiers système sur la pile du niveau 1 à 4 inclus.
#1158Fh : crée un objet graphique vierge de dimensions représentées par les entiers système des niveaux 1 et 2.
#50578h : donne la hauteur et la largeur du grob présent au niveau 1 par 2 entiers système.
#12815h : effectue 1->GROB sur la chaîne présente au niveau 1 et la place aux coordonnées représentées par les entiers système des niveaux 2 et 3 sur le grob du niveau 4, le résultat étant un nouveau grob au niveau 1.
#12E89h : le grob du niveau 1 devient l'objet graphique du grob de la pile, même si celui ci dépasse les dimensions 131*64.

#130CAh : ramène le grob de la pile à ses dimensions standards c'est à dire 131*64.

G) Le clavier.

Il est intéressant de connaître quelques adresses concernant la gestion du clavier, en effet l'appui sur une touche peut être déterminant pour un programme (surtout en mode graphique ; pour les jeux...). Le code de la touche qui est ou a été appuyée est renvoyé sous forme d'entier système. Le code de la touche [A] est par exemple <1h> ainsi de suite en allant ligne par ligne ([G] = <7h>; [L] = <Ch>...).

Quelques adresses:

#420A0h : attend une touche, le résultat est un External TRUE sur le niveau 1 et le code de la touche au niveau 2 ou un booléen FALSE sans code touche.

#4714h : une touche a-t-elle été appuyée ? Le résultat est le même que l'adresse ci dessus.

#D71h : vide le buffer clavier.

42262h : teste si la touche ON a été appuyée la réponse est un booléen.

#5068h : cette adresse permet de mettre à 0 une autre adresse se souvenant du nombre d'appui sur la touche ON.

H) Création de variables locales.

Les variables locales ne servent uniquement que pendant le déroulement d'un programme. On peut stocker n'importe quel objet dans une variable locale , elles sont parfois indispensables mais je vous conseil plutôt de vous servir de la pile pour gagner du temps.

Quelques adresses:

#61CE9h : crée un nombre de variable locale (+ 1) donné par l'entier système du niveau 2 et de même nom donné au niveau 1 la variable n°1 ne doit pas être utilisée donc la variable n°2 reçoit l'objet du niveau 3 avant l'utilisation de cette adresse.

#74E4h : crée un nombre de variable locale donné par l'entier système du niveau 1 (on ajoute 1 a l'entier système avant création).

#75A5h : rappelle le contenu de la variable de numéro donné par l'entier système du niveau 1.

#75E9h : place l'objet du niveau 2 dans la variable locale de numéro donné par l'entier système du niveau 1.

#615E0h : stocke l'objet du niveau 1 dans la variable locale N°1.
#615F0h : stocke l'objet du niveau 1 dans la variable locale N°2.
#61600h : stocke l'objet du niveau 1 dans la variable locale N°3.
#61615h : stocke l'objet du niveau 1 dans la variable locale N°4.
#61625h : stocke l'objet du niveau 1 dans la variable locale N°5.
#61635h : stocke l'objet du niveau 1 dans la variable locale N°6.
#613B6h : rappelle le contenu de la variable locale N°1.
#613E7h : rappelle le contenu de la variable locale N°2.
#614E0h : rappelle le contenu de la variable locale N°3.
#61438h : rappelle le contenu de la variable locale N°4.
#6145Ch : rappelle le contenu de la variable locale N°5.
#6146Ch : rappelle le contenu de la variable locale N°6.

I) La vérification des arguments sur la pile.

Vous aurez sans doute besoin de tester le nombre et le type des arguments nécessaires sur la pile pour que votre programme se déroule sans problèmes ; voici quelques adresses qui éviteront le plantage de votre HP 48 :

0 argument nécessaire : #18A15h ; 1 argument nécessaire : #18AB2h ; 2 : #18A8Dh ; 3 : #18A68h ; 4 : #18B9Fh ; 5 : #18B7Ah. Il faudra aussi tester le type du ou des objets nécessaire(s) à votre programme; l'adresse #18FB2h réalise cette fonction, elle est suivie d'un entier système de préférence en hexadécimal identifiant au maximum 5 objets. Comme en RPL (avec l'instruction TYPE), chaque objet à son identificateur. Les principaux sont <0h> pour un objet quelconque, <1h> pour un réel, <3h> pour une chaîne de caractères, <8h> pour un programme, <Ch> pour un graphique...

Exemple : on veut vérifier la présence d'une chaîne de caractères au niveau 1 et d'un objet graphique au niveau 2 : on procédera de cette manière :

#18AF2h : 2 arguments présents sue la pile sinon erreur.
#18FB2h : vérification des types.
<C3h> : <3h> pour la chaîne au niveau 1 et <Ch> pour l'objet graphique au niveau 2.

Nous allons voir maintenant comment utiliser ces adresses afin de réaliser un programme. Comme pour l'assembleur on utilise une chaîne de caractères pour faire le source, je vous conseil d'utiliser un éditeur de texte tel que le STRING WRITER de Jean Yves AVENARD). Ensuite le fichier source sera transformé en programme exécutable par l'intermédiaire d'un compilateur. Quand je travaille sur ma HP, j'utilise personnellement la DEV.LIB d'Etienne De FORAS qui est la plus simple d'utilisation et propose beaucoup de commandes. (C'est un FREEWARE). La fonction ->S permet de transformer un source en programme exécutable (la fonction S-> réalise le contraire) le $ (alpha shift gauche 4) permet d'indiquer une adresse à transformer le ¤ (alpha shift gauche 6) permet d 'inclure un entier système. La fonction INCLUDE XXX permet d'inclure l'objet qui est contenu dans la variable XXX. La fonction K$ permet d 'inclure un code ASCII sous forme de Character (tout comme l'entier système un Character est directement compréhensible par le microprocesseur du HP 48 on a 0<K$<255 obligatoirement. La fonction ~ (alpha shift droite V) permet de ne pas évaluer l'objet qui suit cette commande (essayez donc de faire ~ DUP et vous comprendrez aussitôt). La fonction ->RAM permet de visionner le contenu d'une commande (la commande peut être par exemple DUP; essayez donc ceci : ~ DUP ->RAM ->S et vous comprendrez pourquoi j'ai dit que les commandes auxquelles vous avez accès sont programmées en External). Voir à ce propos l'exemple en fin d'article pour mieux comprendre quelle allure à un fichier source. Nous allons montrer ici les différentes étapes afin de voir les adresses auxquelles la fonction SWAP fait appel :

~ SWAP *: on fait un NOEVAL sur la fonction SWAP ; SWAP apparaît au niveau 1 de la pile.*

->RAM *: affiche le contenu de la fonction SWAP sous l a forme «External External» au niveau 1.*

->S *: transforme chaque External avec son adresse on obtient:*

PRG *: il s'agit d'un programme.*

$ 18A80 *: Vérifie la présence de 2 arguments, en rappelant du nom de la commande pour le message d'erreur éventuel.*

$ 3223 *: forme interne de SWAP.*

END *: fin de programme.*

Si vous possédez un compatible PC je vous conseille d'aller lire l'encadré en fin d'article.

Voici un exemple de programmation en External. Le but du programme est de faire défiler une chaîne de caractères (présente au niveau 2) en mode graphique à une allure définie par le réel du

niveau 1. Il y a possibilité de stopper le défilement avec la touche [ENTER] et de sortir avec n'importe quelle autre touche. Allez dans votre éditeur de texte HP et entrer le source suivant (les commentaires en italique ne sont pas à mettre, ils servent d'explication) :

PRG *: début du programme.*
$ 18A8D *: vérifie la présence de 2 arguments sur la pile.*
$ 18FB2 *: teste le type des objets présents aux niveaux 1 et 2.*
¤ 31h *: le 1 pour le réel au niveau 1 et le 3 pour une chaîne au niveau 2.*
PRG *: si aucune erreur alors exécute cet objet ici un programme.*
$ 5046A *: efface l 'écran de la pile.*
$ 4E2CF *: efface la barre des menus.*
$ 18CD7 *: transforme le réel du niveau 1 en entier système.*
$ 62747 *: cette réalise SWAP et DUP en interne.*
$ 5636 *: donne la taille de la chaîne sous forme d'un entier système.*
¤ 15h *: ou 21 en décimal...*
$ 3DE0 *: soustrait 2 entiers système; ici enlève 15h à l'entier système indiquant la taille de la chaîne .*
$ 73DB *: FOR i = 1 to ... (entier système obtenu grâce au résultat de l 'adresse $ 3DE0.*
$ 3188 *: forme interne de DUP (duplique la chaîne).*
$ 7221 *: ressort l'indice de la boucle.*
$ 3188 *: forme interne de DUP (duplique l'indice).*
¤ 16h *: ou 22 en décimal.*
$ 3DBC *: on ajoute l'indice + <16h> (taille maximum à afficher en une seule fois).*
$ 5733 *: on effectue un SUB sur la chaîne de caractères du niveau 3 grâce aux entiers système des niveaux 1 et 2.*
$ 11D00 *: on effectue la commande 2->GROB sur ce morceau de chaîne.*
$ 4E712 *: on transforme le grob 132*8 en grob 131*8; on enlève donc la dernière colonne du grob (sans importance).*
$ 122FF *: inversion de ce grob (on fait un petit plus...).*
¤ *0h : ou 0 en décimal !*
¤ 1Ch *: ou 28 en décimal.*
$ 3295 *: forme interne de ROT (le grob revient au niveau 1).*
$ 128B0 *: on place ce grob en coordonnées X=<0h> et Y=<1Ch> grâce aux entiers système .*
$ 32C2 *: forme interne de OVER (on a besoin de l entier système pour l'allure de l'affichage.*
$ 73DB *: FOR i = 1 to ....*
$ 40EE7 *: attends environ 5/100 ème de seconde.*
$ 7334 *: next on revient au FOR i ci dessus.*
$ 4708 *: teste si une touche a été appuyée .*
$ 619BC *: IF THEN (if booléen = TRUE THEN).*
PRG *: alors on exécute encore un programme.*
¤ 19h *: ou 25 en décimal ; code de la touche [ENTER].*
$ 3D19h *: teste si 2 entiers système sont égaux c'est à dire s'il s'agit de la touche [ENTER].*
$ 61AD8 *: IF TRUE OBJ1 ELSE OBJ2 END.*
PRG *: ...(objet TRUE).*
$ D71h *: vide le buffer clavier.*
$ 420A0h *: attends l'appui sur une touche.*
$ 619BC *: si le booléen est true alors.*
$ 3244 *: effectue un DROP car on a pas besoin du code de la touche qui permet de refaire défiler la chaîne.*
END *: fin programme donc fin de OBJ1.*
$ 6400F *: au prochain next on sort...c 'est l'objet FALSE on aurait pu mettre à la place PRG $ 6400F END.*
END *: fin programme a la suite du teste du buffer clavier $ 4708.*
$ 7334 *: next on affiche un caractère de plus.*
$ 40F02 *: on attends un peu.*
$ 60F9B : on exécute un DROP sur le niveau 2.
$ D71 *: on vide le buffer clavier.*
END *: .. fin de l 'objet exécuté si les arguments étaient les bons.*
END *: .. fin du programme.*

H) Conseils :

Je vous conseille de réaliser toutes vos sources sur papier avant de les copier sur votre éditeur de texte et ensuite de les compiler. Vérifiez bien si vous ne vous êtes pas trompé en recopiant les adresses. (Ex : ne pas mettre un F à la place d'un 7 dans une adresse...) et surtout vérifiez bien les arguments nécessaires avant d'appeler une adresse sinon vous allez au pire vers le message désagréable « Memory Clear «. Ne confondez pas $ et le ¤, ceci est fréquent. L'adresse #6657h (Forme Interne de NEWOB) est très importante : si vous avez par exemple une variable qui contient un graphique et que vous appelez le contenu de cette variable sur le niveau 1 de la pile, je vous conseille d'appliquer l'adresse #6657h car certaines adresses n'effectuent pas de NEWOB avant d'agir. Il en résulte une transformation directe au sein de la variable.

PROGRAMME PCDEV
pour PC compatibles

Il s'agit d'un programme permettant de programmer directement en external sur votre PC. Vous avez la possibilité de donner un nom (ou mnémonique) à chaque adresse. Vous pouvez compiler directement sur votre PC en tenant compte de la série de votre calculatrice (S(x) ou G(x)). On peut appeler une adresse avec $, inclure un objet HP (graphique, programme...) avec la fonction INCLUDE (si cet objet n'est pas sur l'ordinateur, PCDEV va directement le chercher sur la HP.). Il y a gestion des XLIB pour la G(x) et surtout il y a la possibilité de gérer les mnémoniques (recherche par adresse, par mnémonique), avec un commentaire expliquant chaque adresse...

Demandez une version d'évaluation (précisez si vous désirez une disquette 720K ou 1,44M) et envoyez 30F en timbres à :

POCHE COMPUTERS
programme PCDEV
42220 -BURDIGNES - FRANCE

INITIATION A L'ASSEMBLEUR SUR HP 48

(2ème partie faisant suite au N°2 de POCHE COMPUTERS par Régis DUCHESNE)

(4.3) Instructions de branchements

Dans un source, puisqu'il n'existe pas de numéro de ligne comme en BASIC standard, on déclare un label, c'est-à-dire un point où l'on désirerait aller, par exemple pour effectuer une boucle, par le symbole *, immédiatement suivi du label. Ce saut est un saut dit relatif. Par contre, si vous désirez sauter à un endroit précis de la mémoire, pour allez éxécuter un sous-programme qui existe déjà en ROM (il ne faut pas oublier que votre machine sait déjà faire beaucoup de choses lorsque vous l'achetez...), il vous faudra utiliser un saut dit absolu.

a) Sauts inconditionnels

GOTO label %Saute à la première instruction qui suit le label

Remarque: Il se peut que, lors de l'assemblage de votre source, ASMFLASH vous signale le message d'erreur «Jump Too Big». Dans ce cas, remplacez cette instruction par une instruction faisant exactement la même chose, mais qui est codée différemment de manière interne par la HP: GOLONG label.

GOVLNG adresse %Saute à l'adresse mémoire adresse, qui doit forcément correspondre à l'adresse d'une instruction d'un programme en langage machine.

Remarque pratique: Pour rendre la main au système, plutôt que de taper les fastidieuses trois instructions A=DAT0 A, D0=D0+ 5, PC=(A), on peut écrire GOVLNG 12002. En effet, si l'on utilise un désassembleur (c'est-à-dire un programme faisant l'opération inverse, lorsqu'elle est possible, d'un assembleur, j'utilise personnellement UNAS version 1.5 de FRENHOFER, que vous trouverez sans problème sur le serveur minitel TOP JEF), on constate que se trouve à l'adresse 12002 en ROM (toutes versions de HP48 confondues), un petit bout de langage machine dont le source correspondant est exactement constitué de ces trois instructions.

b) Sous programmes

GOSUB label %Saute à la première instruction du sous-programme qui suit le label, et revient à l'instruction qui suit immédiatement le GOSUB dès qu'une instruction de retour (cf fin du paragraphe) à été rencontrée dans le sous-programme.

Remarque: Il se peut que, lors de l'assemblage de votre source, ASMFLASH vous signale le message d'erreur «Jump Too Big». Dans ce cas, remplacez cette instruction par une instruction faisant exactement la même chose, mais qui est codée différemment de manière interne par la HP: GOSUBL label.

GOSBVL adresse %Saute à l'adresse mémoire adresse, qui doit forcément correspondre à l'adresse d'un sous-programme en langage machine, et revient à l'instruction qui suit immédiatement le GOSBVL dès qu'une instruction de retour (cf fin du paragraphe) a été rencontrée dans le sous-programme.

Remarque pratique: Pour éviter d'avoir à faire attention, lors de l'écriture d'un programme, aux registres utilisés par la machine et que l'on ne doit pas modifier, on peut utiliser deux sous-programmes très sympathiques: l'un au début du programme, par l'instruction GOSBVL 0679B, qui stocke B A, D A, D0, D1 à un endroit précis de la mémoire, et l'autre en fin de programme, par l'instruction GOSBVL 067D2, qui récupère les valeurs initiales de ces registres en allant les rechercher là où elles ont été stockées. (Le désassembleur qui permet de trouver ces adresses intéressantes est un outil extrêmement utile.)

Instructions de retour:

Retours inconditionnels:

RTN %Retour simple

RTNCC %Retour en désarmant la CARRY

RTNSC %Retour en armant la CARRY

Retours conditionnels:

RTNC %Retour uniquement si la CARRY est armée

RTNNC %Retour uniquement si la CARRY est désarmée.

c) Instructions de tests

GOC label %Saute au label spécifié comme avec un GOTO, mais uniquement si la CARRY est armée.

GONC label %Idem si la CARRY est désarmée.

Exemple: Voici le source typique d'une boucle de temporisation:

«LC 20000 Chargement de C A à une valeur qui correspond sur les HP48S(X) à une attente d'environ 4 sec et sur les HP48G(X) 2 sec.

*BOUCLE %Déclaration de label pour pouvoir revenir au début de la boucle

C=C-1 A Décrémente C A. Si C A ne valait pas 0, la CARRY reste désarmée. Par contre, si C A valait 0, il y a retenue donc la CARRY s'arme.

GONC BOUCLE %Si la CARRY est désarmé, on retourne au début de la boucle, sinon, on sort de la boucle.

GOVLNG 12002 %Rend la main au système.

@»

Exécutez-le après l'avoir assemblé, vous apprécierez la rapidité du langage machine qui vous permet d'éxécuter 20000 (soit 131072 en décimal) boucles en 2 secondes...

De manière générale, un test s'écrit sur deux lignes d'instructions : la première ligne contient le test à proprement parler, la seconde contient ce qu'il faut faire si le test s'avère concluant.

Instructions de première ligne:

?A=0 f %A champ f vaut-il zéro?

Remarque: Cette dernière instruction existe aussi en remplaçant A par B, C, ou D

?ABIT=0 x %Le bit numéro x de A est-il nul?

?ABIT=1 x %Le bit numéro x de A est-il à 1?

Remarque: Ces deux dernières instructions existent aussi en remplaçant A par C

?P= x %P a-t-il la valeur x?

?P# x %P est-il différent de la valeur x?

?ST=0 x %le drapeau x est-il désarmé?

?ST=1 x %le drapeau x est-il armé?

?A=B f %les champs f de A et de B sont-ils égaux?

?A=C f %les champs f de A et de C sont-ils égaux?

?B=C f %les champs f de B et de C sont-ils égaux?

?C=A f %les champs f de C et de A sont-ils égaux?

?C=B f %les champs f de C et de B sont-

INITIATION A L'ASSEMBLEUR SUR HP 48

ils égaux?
?C=D f %les champs f de C et de D sont-ils égaux?
?D=C f %les champs f de D et de C sont-ils égaux?
?A#B f %les champs f de A et de B sont-ils différents?
?A#C f %les champs f de A et de C sont-ils différents?
?B#C f %les champs f de B et de C sont-ils différents?
?C#A f %les champs f de C et de A sont-ils différents?
?C#B f %les champs f de C et de B sont-ils différents?
?C#D f %les champs f de C et de D sont-ils différents?
?D#C f %les champs f de D et de C sont-ils différents?
?A<=B f %A champ f est-il inférieur ou égal à B champ f?
?A<B f %A champ f est-il strictement inférieur à B champ f?
?A>=B f %A champ f est-il supérieur ou égal à B champ f?
?A>B f %A champ f est-il strictement supérieur à B champ f?
?B<=C f %B champ f est-il inférieur ou égal à C champ f?
?B<C f %B champ f est-il strictement inférieur à C champ f?
?B>=C f %B champ f est-il supérieur ou égal à C champ f?
?B>C f %B champ f est-il strictement supérieur à C champ f?
?C<=A f %C champ f est-il inférieur ou égal à A champ f?
?C<A f %C champ f est-il strictement inférieur à A champ f?
?C>=A f %C champ f est-il supérieur ou égal à A champ f?
?C>A f %C champ f est-il strictement supérieur à A champ f?
?D<=C f %D champ f est-il inférieur ou égal à C champ f?
?D<C f %D champ f est-il strictement inférieur à C champ f?
?D>=C f %D champ f est-il supérieur ou égal à C champ f?
?D>C f %D champ f est-il strictement supérieur à C champ f?
Instructions de seconde ligne:
GOYES label %Si le test est vérifiée, se branche à la première instruction suivant le label spécifié.
RTNYES %Si le test est vérifié, sort du sous-programme (c'est une instruction de retour).

Remarque utile: Si lors de l'assemblage d'un source vous obtenez le message d'erreur «Jump Too Big» dans un morceau de source de ce type:
...source avant le test
?A=0 A
GOYES NATHALIE
...source après le test
vous devez, pour pouvoir correctement l'assembler, le modifier de la manière suivante:
...source avant le test
?A#0 A
GOYES BRISSET
GOTO NATHALIE
*BRISSET
...source après le test

On peut ainsi remplacer n'importe quel test par son contraire...

(5) Quelques exemples commentés

Inversion vidéo de l'écran

«GOSBVL 0679B %Sauvegarde des registres utilisés par la machine
D0=8068D %adresse où est stockée l'adresse de l'écran sur 5 quartets, pour HP48G(X) %à remplacer par 7050E pour une HP48S(X)
C=DAT0 A %C A=adresse de l'écran
D0=C %D0=adresse de l'écranLC 37 %boucle 38 (56 en décimal) fois, pour passer sur toutes les lignes de l'écran.*BOUCLE %Une ligne fait 34 quartets de long, soit 16+16+2 (tout en décimal)
A=DAT0 W %Lit 16 quartetsA=-A-1 W %Opérations logique NOT: inversion vidéo
DATO=A W %Réécrit les 16 (décimal) quartets
D0=D0+ 16 %Avance le pointeur écran pour traiter les quartets suivants
A=DAT0 W
A=-A-1 WDATO=A W %Recommence avec les 16 (décimal) quartets suivants
D0=D0+ 16
A=DAT0 B
A=-A-1 BDATO=A B %Termine avec les 2 quartets restants
D0=D0+ 2
C=C-1 B % Une ligne de moins à traiter
GONC BOUCLE % Si ce n'était pas la dernière ligne, recommence la boucle.
GOSBVL 067D2 %Récupère les registres utilisés par la machine
GOVLNG 12002 %Rend la main au système
@»

Assemblez-le et exécutez-le, vous serez encore une fois surpris par l'instantanéité du phénomène langage machine...

Identicité de deux objets

Ecrivons un programme permettant de tester si deux objets placés aux niveaux 1 et 2 de la pile sont les mêmes: il supprime ses deux arguments et renvoie la chaîne «OUI» si les objets sont les mêmes et «NON» s'ils sont différents.
La grosse ruse employée dans ce programme est que lors d'un GOSUB, l'adresse de retour, c'est-à-dire l'adresse suivant immédiatement le gosub en mémoire, est stockée dans le dernier étage de la pile RSTK. Cela permet à la machine, lors d'une instruction de retour, de savoir où elle doit revenir, en dépilant la valeur de retour. Par notre dépilage manuel avec l'instruction C=RSTK, on récupère l'adresse où sont stockées les chaines «OUI» et «NON», ce qui permet de les placer sur la pile...
«GOSBVL 0679B
%Récupère les adresses des objets
C=DAT1 A %C=adresse de l'objet au niveau 1 de la pile
R0=C A %R0 A=adresse de l'objet au niveau 1 de la pile
GOSUB TAILLE %Calcule la taille de l'objet au niveau 1 de la pile
B=C A %Stocke la taille de l'objet du niveau 1 de la pile
D1=D1+ 5 %D1 qui n'a pas été modifié pointe maintenant sur le niveau 2 de la pile
C=DAT1 A %A=adresse de l'objet au niveau 2 de la pile
R1=C A %R1 C=adresse de l'objet au niveau 2 de la pile
GOSUB TAILLE %Calcule la taille de l'objet au niveau 2 de la pile
?B=C A %Comparaison des tailles des objets
GOYES OK %Sinon les objets sont forcéments différents, on le sait
GOTO DITNON
%Comparaison des deux objets:
C=R0 A %Récupère l'adresse de l'objet au niveau 1
D0=C A % DO pointe sur l'objet du niveau 1
C=R1 A
D1=C A %Fait de même avec D1 pour l'objet du niveau 2
*BOUCLE
C=DAT0 1 %Lit un quartet de l'objet du niveau 1
D0=D0+ 1 %avance son pointeur pour pouvoir lire le quartet suivant plus tard
A=DAT1 1
D1=D1+ 1 %Fait de même pour l'objet du niveau 2
?C=A P %Test l'égalité des quartets lus

(P= 0 par défaut...)
GOYES OK2 %S' ils sont différents, les objets le sont aussi, donc:
GOTO DITNON
*OK2
B=B-1 A %On vient de traiter un quartet de plus
?B#0 A %Reste-t-il des quartets?
GOYES BOUCLE %Si oui, on continue la vérification
%Sinon, les deux objets sont forcément les mêmes
GOSUB OUI
$C2A20B0000F45594 %Correspond à la représentation interne de la chaîne «OUI»
*OUI
C=RSTK %Récupère l'adresse de la chaine «NON»
GOTO FIN %Va la mettre sur la pile...
*DITNON
GOSUB NON
$C2A20B0000E4F4E4 %Correspond à la représentation interne de la chaîne «NON»
*NON
C=RSTK %Récupère l'adresse de la chaine «NON»
*FIN
A=C A %Stocke l'adresse C A dans A A
GOSBVL 067D2 %Récupère les registres de la machine (modifie C A et pas A A)
D1=D1+ 5 %Effectue un DROP en disant à la machine qu'à partir de maintenant, l'objet
%du niveau 1 n'existe plus
D=D+1 A %Augmente le registre de mémoire libre de 5, puisqu'un niveau donc une adresse
%a été supprimée, et qu'une adresse est codée sur 5 quartets.
DAT1=A A %Place la chaîne «OUI» ou «NON» sur le nouveau niveau 1, ex-niveau 2
GOVLNG 12002 %Rend la main au système
*TAILLE
%Sous-programme qui calcule la taille d'un objets pointé par C champ A
%Remarque: ce sous-programme ne modifie pas D1 et B
D0=C %D0=adresse de l'objet dont on veut connaître l'adresse de fin
RSTK=C %Stockage temporaire de l'adresse de début de l'objet
GOSBVL 03019 %Calcul l'adresse de fin d'un objet dont l'adresse de début est dans D0
%l'adresse de fin est renvoyée dans D0 (merci le désassembleur...)
AD0EX %A=adresse de fin de l'objet
C=RSTK %Récupère l'adresse de début de l'objet
C=C-A A %C=taille de l'objet en quartets
RTN
@»

Avant d'assembler ce programme, n'oubliez pas de mettre deux objets sur la pile, ça peut toujours servir...

(6) Ruses à employer

Compatibilité S et G

Vous êtes les futurs programmeurs qui allez assurer la relève des «vieux» que nous sommes. Alors autant que vous preniez de bonnes habitudes tout de suite:

Pour assurer une compatibilité des HP48S(X) et des HP48G(X), il faut garder à l'esprit que certaines adresses ont changé de place, mais pas toutes. En fait ce sont beaucoup les adresses de sauvegarde qui ont bougé de la S à la G, en gros tout ce qui était en 70000 sur la S est passé en 80000 sur la G Les adresses des sous-programmes, vous avez pu le constater dans les exemples, n'ont elles, pas bougé.

Alors que faire? Ecrire une table dans son programme, table que l'on va consulter en fonction de la machine sur laquelle on fonctionne? Eh bien non, il y a beaucoup plus simple, en fait, aussi incroyable que cela puisse paraître, les machines contiennent cette table de conversion!!!

Exemple: Vous avez pu constater dans l'exemple d'inversion vidéo de l'écran que l'adresse de sauvegarde de l'adresse de début d'écran n'étaient pas les mêmes sur S et sur G. Vu que des routines dans la ROM de la HP font référence à cette adresse de sauvegarde, et que ces routines n'ont pas bougé (c'est uniquement l'adresse à laquelle elles font référence qui a changé), pourquoi ne pas aller lire cette référence à cette adresse de sauvegarde directement au sein de ces routines? Il n'ya plus qu'a : voici un morceau de source à bien retenir:
D0= 131F8 %adresse d'une référence à l'adresse de sauvegarde commune aux HP48S(X)
%et aux HP48G(X)
C=DAT0 A %C=7050E sur S et 8068D sur G, le tour est joué!
puis le source continue comme tout à l'heure:
D0=C %D0 pointe sur la sauvegarde de l'adresse écran
C=DAT0 A %C=adresse de l'écran

En gros, pour assurer la compatibilité, on écrit deux lignes de plus. Que sont ces deux lignes éxécutées en une fraction de seconde grâce au langage machine, comparé au confort d'utilisation d'un programme compatible sur deux machines différentes?

Le GOSUB sans retour

Entre programmeurs, il existe des concours de programmation (par exemple l'on doit écrire un sous-programme en langage assembleur qui trace une ligne)

Ces concours sont basés sur deux critères de sélection: la vitesse d'éxécution et la taille du sous-programme.

Exemple : Voici le sous-programme écrit par HPMAD permettant de calculer dans C W la racine carré de A W:
C=0 W
C=C-1 W
*LOOP
C=C+2 W
A=A-C W
GONC LOOP
CSRB W

je vous invite à vérifier que ce sous-programme fait bien ce qu'il doit faire, remarquez au passage qu'il ne fait que 11,5 octets. Par contre, il est très lent sur les grands nombres...

Revenons à nos moutons, je vais vous donner des petits trucs bien utiles pour gagner en vitesse et en taille mémoire dans la majorité de vos programmes:
- On ne met jamais de RTN après un appel de sous-programme:
Le source suivant:
...source avant l'appel
GOSUB SSPRG
...source après l'appel
*SSPRG
...début du sous-programme
GOSUB GAG
RTN
est rigoureusement équivalent à celui-ci:
...source avant l'appel
GOSUB SSPRG
...source après l'appel
*SSPRG
...début du sous-programme
GOTO GAG

Le pourquoi du comment est expliqué au début de l'exemple portant sur l'identicité de deux objets. Procédez pas à pas, vous comprendrez.
- Utilisez préférentiellement l'instruction LC à LA, elle est plus rapide et prend moins de taille mémoire.
- Utilisez des GOC et des GONC plutôt

INITIATION A L'ASSEMBLEUR SUR HP 48

que des GOTO lorsque vous connaissez l'état de la CARRY au moment d'un saut, ces instructions sont plus rapides et prennent moins de taille mémoire.

(7) Conclusion

Vous voilà arrivé au terme de votre initiation, je suis certain que vous vous posez encore beaucoup d'autres interrogations au sujet de ce magnifique langage, qui permet, non sans mal il est vrai, de tout faire puisqu'il est rapide et qu'on accède grâce à lui directement au coeur de la machine.

Je n'ai plus qu'un conseil: courrez commander «Voyage au centre de la HP48», son prix (300 F) un rien déconcertant au début est loin de sa valeur réelle. Ce livre recèle des secrets qui vous permettront, en y mettant un peu du vôtre, d'exploiter votre HP à son maximum, par exemple de protéger ou déprotéger des cartes sans arrêts systême, de construire des objets définis par vous-même, de faire du tracé en trois dimensions en temps réel, de transmmettre des données à 11200 bauds, de télécommander votre télévision avec la HP, que sais-je encore!

Lorsque vous constaterez qu'autant de puissance sommeille dans cette machine, vous comprendrez pourquoi son principal intérêt est qu'elle tient dans la poche...

CABLE DE TRANSMISSION HP48 <----> PC

par Mickaël SAILLOT

1°) Le câble de transmission est constitué de trois parties:

- Un connecteur 4 broches espacées de 2mm pour la calculatrice.
- Un réseau de fil électrique.
- Un connecteur 9 ou 25 PINS pour l'ordinateur.

2°) Le connecteur de la HP48

```
Connecteur  Diodes IR
 o o o o     O  O
 1 2 3 4
```

3 : Broche de réception des données
2 : Broche de transmission des données
4 : Masse logique
1 : Masse mécanique

3°) HP48 / 9 PIN

a) Numérotation des broches sur un connecteur 9 PIN:

```
1 2 3 4 5
o o o o o
 o o o o
 6 7 8 9
```

b) Connections:

```
HP          9-PINS
3 ———————— 3
2 ———————— 2
4 ———————— 5
1 ———————— Blindage du connecteur
```

4°) HP48 / 25 PINS

a) Numérotation des broches sur un connecteur 25 PIN:

```
1  2  3  4  5  6  7  8  9  10 11 12 13
o  o  o  o  o  o  o  o  o  o  o  o  o
 o  o  o  o  o  o  o  o  o  o  o  o
14 15 16 17 18 19 20 21 22 23 24 25
```

b) Connections:

```
HP          25-PINS
3 ———————— 2
2 ———————— 3
4 ———————— 7
1 ———————— Blindage du connecteur
```

5°) La HP48 possède une interface de transmission série asynchrone,une interface UART (Universel Asynchronous Receiver Transmitter), qui lui permet de communiquer avec toutes les machines possédant une interface série RS232.

Cela permet de commander les systèmes électroniques ayant une interface RS232 à condition d'avoir correctement configuré le port série de la calculatrice (voir SETUP,IOPAR,PRTPAR) en accord avec le système de transmission de l'appareil.

Voyons comment il est possible de transmettre des données:

Il faut définir d'abord 3 états logiques:

- l'état 1 qui correspond à une tension négative.
- l'état 0 qui correspond à une tension positive.
- l'état X qui correspond a une tension nulle.

Pour envoyer un octet,on commence par envoyer 1 bit de départ (0), puis les données et enfin 2 bits de stop (1).

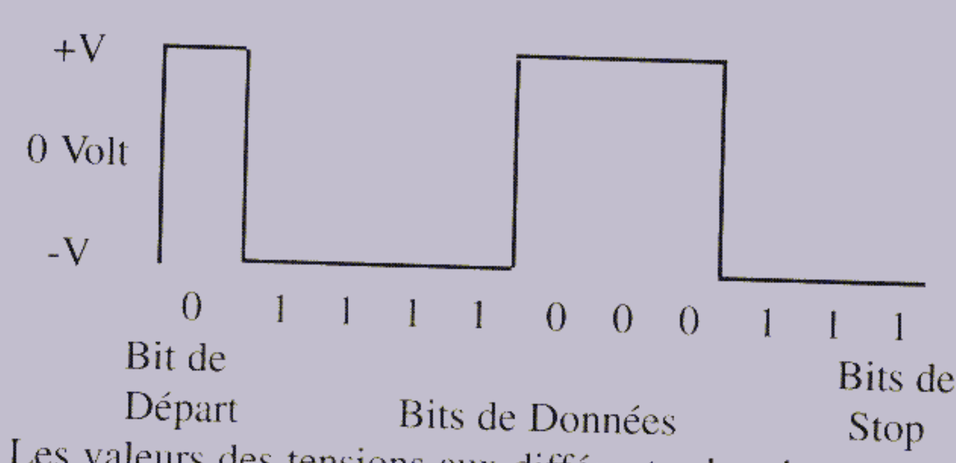
Voyons cela sur un exemple :

Si on veut envoyer l'octet #8Fh:

#8Fh s'ecrit 10001111 en binaire , on envoit donc

- 1 bit de départ 0
- 8 bits de données 1 1 1 1 0 0 0 1 (On va du bit le plus bas au bit le plus fort)
- 2 bits de stop 1 1

D'un point de vue électrique on aura :



Les valeurs des tensions aux différentes broches :

-Sur la broche transmission, la tension est de +/- 3.0 V selon l'état (1 ou 0). La tolérance est 2.5%.

-Sur la broche de réception, la HP percoit un état 0 si la tension est comprise entre +1.0 V et +15.0 V ; elle perçoit un état 1 si la tension est comprise entre -15.0 V et +0.3 V.

Malgrè la marge importante obtenue pour ces tensions les variations entre chaque bit doivent être inférieures à 2.5%.

L'impédance sur la broche de réception doit être comprise entre 5.0 kOhms et 7.0 kOhms avec une tolérance de 2.5%.

Enfin, question logiciel, il y a l'excellent soft de communication KERMIT tombé maintenant dans le domaine public et que vous pouvez acquérir auprès de MICROCARDS (30F en timbres, disquette et port compris) ou gratuitement sur les serveurs.

LIAISON HP 48 <-----> MINITEL

A la demande de nombreux lecteurs, nous publions ici le schéma de liaison entre une HP48 et un minitel. Amis HPistes, à vos fers à souder !!!

Prise minitel (face)

2
4 5
1 3

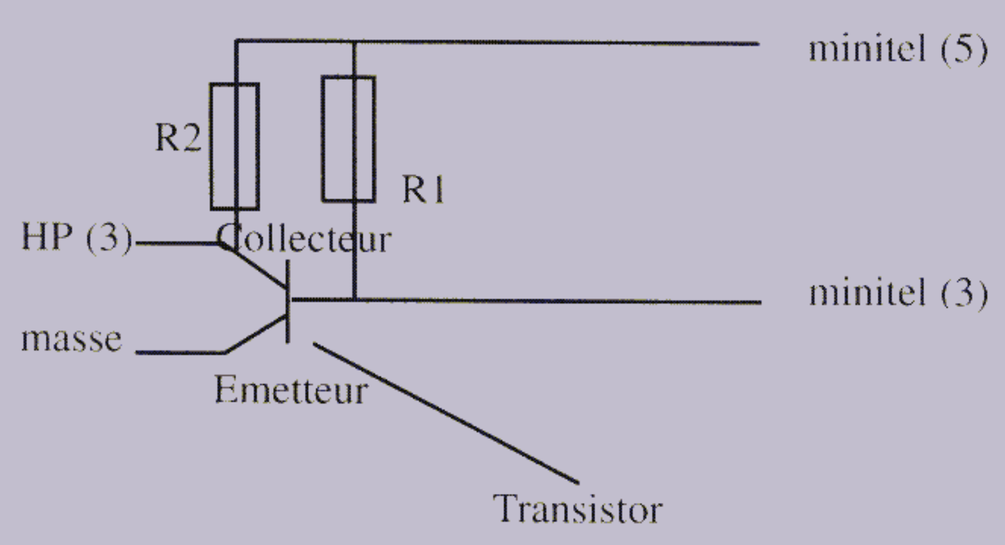
1) Réception par minitel
2) Masse signaux
3) Emission
4) Périphérique (inutilisé ici)
5) Alimentation

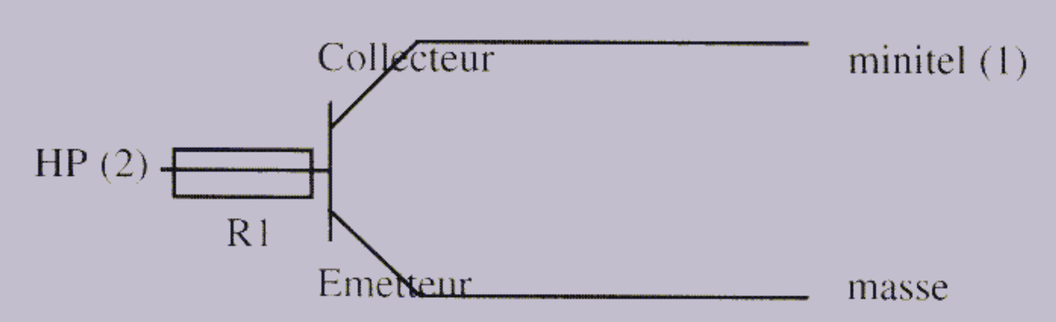
Prise HP (face)

1 2 3 4

1) Masse châssis
2) Transmission par HP
3) Réception
4) Masse signaux

L'opération de câblage nécessite 2 résistances de valeur 47K ohms, notées R1 sur le schéma, une résistance de 27K ohms notée R2 et 2 transistors de type 1711 ou 2222 ou équivalent.





La seule petite difficulté sera de vous procurer le connecteur 4 points pour la HP. A titre d'information, on trouve ce type de connecteurs chez des importateurs en produits électroniques. Sinon, il vous restera la solution d'en "bricoler" un avec du connecteur plus standard (voir votre marchand d'électronique).
Autre solution, commander le câble tout monté et testé chez MICROCARDS (disponibles à partir de juillet au prix de 245F pièce avec soft sur disquette)

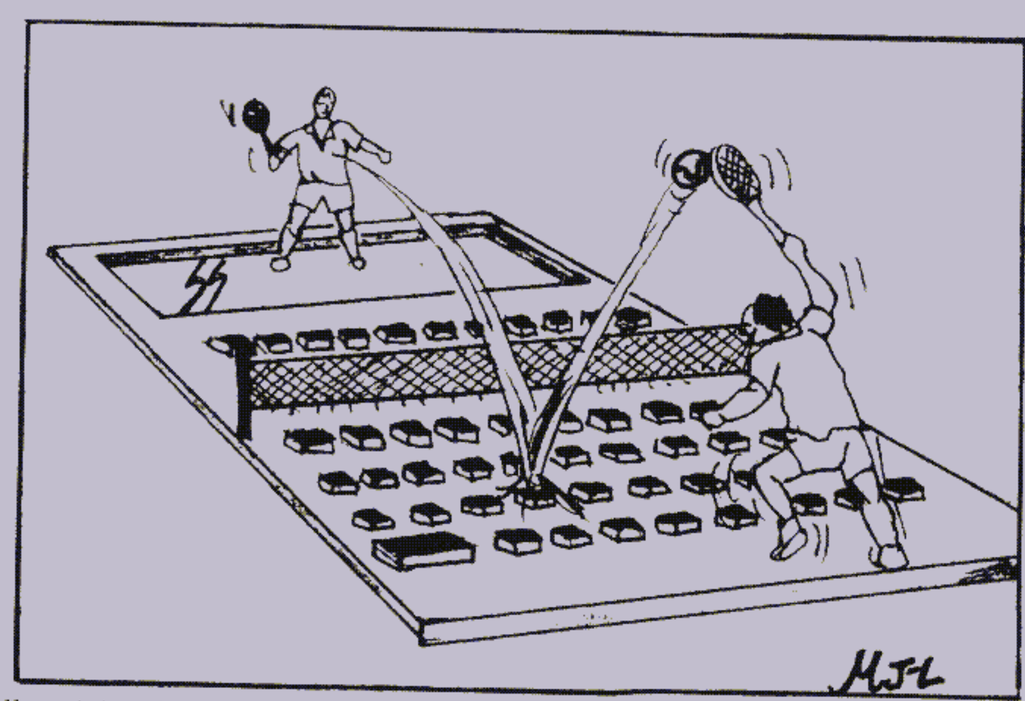
Pocket Computer BBS

Le board entièrement bénévole spécialisé dans les pockets.
SHARP PCE500, HP48, PSION, Laptops ...
(33 1) 42.08.60.82
8 bits sans parité, 1 bit de stop, V21,V22,V22b,V23,V32,V32b,ANSIBBS disponible
SysOp : Guillaume BINET

JEU DE TENNIS SUR HP 28S

(BIARBALL)

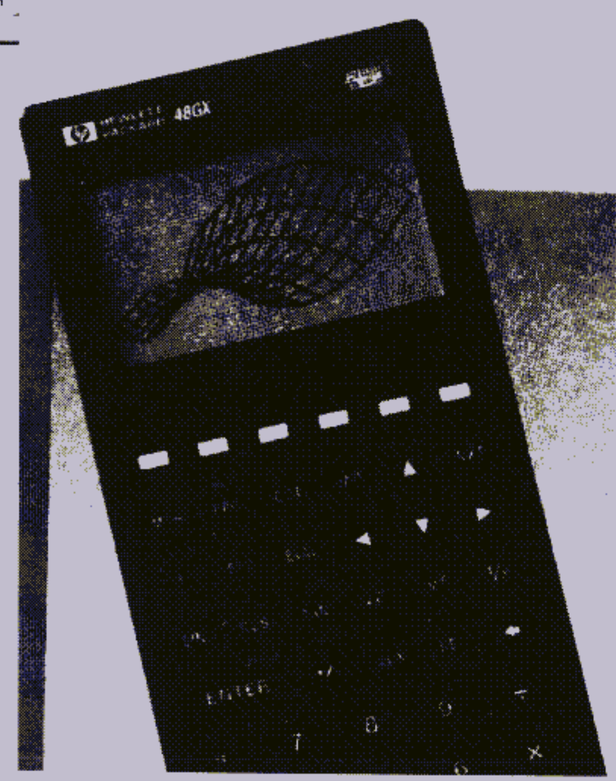
Ce jeu de moins de 1K se joue à deux. Pour jouer, on retourne le clavier de gauche sous la machine. Les touches des mouvements de raquette du joueur de gauche sont INS pour monter et CUR (la touche à gauche de TRIG) pour descendre. Celles du joueur de droite sont RIGHT (la touche au dessus de NEXT) pour monter et USER pour descendre. L'activation ou la désactivation des effets sonores dépend du flag 51, comme en utilisation normale de la machine.
Après avoir rentré la vitesse de jeu (0=très rapide 1=rapide 2=moyen 3=lent 4=très lent) et avoir sélectionné le programme BRDBL, c'est parti !!! Au début, le joueur qui est au service est choisi aléatoirement. Pour servir, le joueur doit monter et descendre sa raquette simultanément. Le but du jeu est de toujours renvoyer la balle (sur le plat de la raquette=rebond long, sur un côté de la raqette=rebond court) contre les bords inférieur et supérieur de l'écran afin de faire perdre son adversaire. celui qui perd une balle a le service pour le coup suivant. La partie se joue en 7 points.
Pour ne pas faire d'erreurs fatales (risques de Memory Lost) en entrant ce programme, bien suivre les étapes successives. Si vous avez déjà rentré les programmes ASSR et CHKR lors de la saisie d'un autre jeu, passer à l'étape V.



I — CREATION DU PROGRAMME ASSL
Rentrez ce programme SANS ERREUR:
<< -> C << HEX «» 1 C SIZE FOR X «#» C X 1 + DUP SUB C X X SUB+ + STR-> B->R CHR + 2 STEP 2 # 8253h SYSEVAL #4F3Dh SYSEVAL >> >>
Stockez-le dans ASSL.
II — CREATION DU PROGRAMME CHKL
Rentrez ce programme SANS ERREUR:
<< -> C << HEX 64 STWS #0h 1 C SIZE FOR X «#» C X DUP SUB + STR-> X # + NEXT >> >>
Stockez-le dans CHKL.
III — CREATION DU PROGAMME ASSR
Rentrez la chaîne suivante SANS ESPACE NI SAUT DE LIGNE:
«
76C20 0C8E1 1E3C0 69C20 F6000 8F180 50147 068F7 2F40A F2306 EE474 81E06 E68F9 38200 7D507 13606 13516 914A3 103EA 31909 EA703 07EA1 59016 1170C D5CD8 DDEDD 109F2 0
«
Dupliquez la chaîne et vérifiez-la avec CHKL. Si le résultat est différent de #E016h, elle a été mal rentrée, recommencez cette étape. Sinon, éxécutez ASSL et stockez le résultat (les 3 System Object) dans ASSR. Ensuite, purgez ASSL.
IV — CREATION DU PROGRAMME CHKR
Rentrez la chaine suivante SANS ESPACE NI SAUT DE LIGNE:
«
76C20 07971 CD570 EEE80 01670 69C20 4B000 8F180 50143 13117 4AF0A F3101 10214 3174D 2305E A81C1 0014B 3103B 6A319 09EA8 0307B 6AAE6 AE711 1E410 1AF13 1F3A7 5A775 50A78 A6E50 F112A 70102 17111 11188 A6BA8 F8BA5 0174E 71431 33179 11A15 57131 14216 4808C 09F20
«
Dupliquez la chaine et vérifiez-la avec CHKL. Si le résultat est différent de #1D0FBh, elle a été mal rentrée, recommencez cette étape. Sinon, éxécutez ASSR et stockez le résultat (le #0h DUP + SWAP System Object) dans CHKR. Ensuite, purger CHKL.
V — CREATION DU PROGRAMME BRDBL
Rentrez les chaines suivantes SANS ESPACE NI SAUT DE LIGNE. Après chacune des saisies de chaîne, il faut dupliquer la dernière chaîne saisie et la vérifier avec CHKR. Si le résultat n'est pas celui indiqué, il faut retaper la chaîne. A la fin, vous devriez normalement vous retrouver avec 8 chaînes dans la pile.
«
76C20 79CB0 D78E0 73D20 10650 D9E04 B211D D0A0E 4A203 3000A 2A2A2 A2A2A 20224 94142 54424 14C4C 402A2 A2A2A 2A2A2 A5832 E4A20 39000 44553 48454 35E45 40202 02020 20202 2303F 20383 F2931 3A025 54749 43502 45341 30202 02020 20202 02020 26323 9393A 0C495 34545 402D4 94C49 44514 94255 40202 3547D 23495 25ED2 116A5 A0803 11A37 A04B2 114B2 113CO A0ED2 11670 9079C BO535 E073D 20106 50167 03COA OED21 16709 079CB 03C0A 0ED21 16709 079CB 0E57A 069C2 0B840 07F52 8418A A5085 11747 D4284 38AA5 0 « CHKR: #6361Eh
85317 47B32 8428A A5085 21747 92210 C145E 7E7E7 8F180 501B5 300C3 0F15C 01B04 8FF1F 02EFF D2233 1FF7F F131E 777F1 31C37 FE1D2 2030C 10910 A30E1 0B167 1E878 F3128 87221 1A8ED F1E00 EF318 1D713 61341 08340 00817 4A134 000C3 79913 10C8F 1DA10 0EF78 A7EE3 40010 0 « CHKR: #2045Dh
CE5DF 32200 AB731 088F1 DA10D50EF3 93BD1 1118A 8511B 048FF 85475 51CC1 01320 10AB7 D90EF 393BD 11128 A8511 B01EF F8547 721CC 10232 080AB 73104 8F1DA 10D50 EF393 B1211 13181 96251 1B048 FF844 7AE0E 41013 2800A B7D90 EF393 B1211 23181 « CHKR: #1A21Ch
96251 1B01E FF844 78B0E 41028 F8B05 0147C E4901 456B0 F11C1 45118 134AF 03130 15A7A 74861 8081C 81C15 87762 2A6E5 2E113 E4861 60CCC C1038 AC908 416F0 031C1 96641 85131 101B1 00007 C5076 90863 607D8 06F8E 14313 31791 47131 « CHKR#15561h
0115C 415D4 1671C 701AF 2302A E715E 7A768 64808 1E81E 15C71 67A6F 53E01 D71FB 110C1 5B030 40EF6 90C00 13232 00813 08011 805CF 32004 801CF 96F2E 01313 0D711 01308 62E37 93113 21301 007C2 17821 13213 01317 B1115 B7158 7 « CHKR #11CD9h
75317 131A6 F5FD6 73073 21132 13010 070F0 13213 01317 90115 B7158 777D0 73D0A 6F5FD AF215 D7110 34858 FF8A6 C0111 85263 10320 FD8A6 00112 84211 BEA87 13634 CFFFF 8B290 85362 70D23 078B2 73073 1C31B 5D000 79DE8 F8B05 0344B 21186 26031 9C145 « #1B805h
14216 4808C 3028B 24B84 36420 34BFF FF8BE 1A306 8BE7E D2E68 BE1A6 D8F31 901B0 40007 67E01 D5348 5AFF1 328A6 90328 FBDAD 2308C A132D 901D5 3400C FF132 8A690 3206A DAD23 08EA1 32D90 12E3E 076C2 009F2 08F3E 076C2 09C21 1EEE8 0 « CHKR: #1B38Fh
F7971 09F20 954E0 76C20 01670 9C211 EEE80 01670 07971 09F20 974E0 3F211 5E670 3F211 5E670 EA970 09BB1 EA970 13AB1 9C211 A37A0 555EO 3F211 5E670 3F211 5E670 5FC90 74311 5D980 D4AE0 44670 E4A20 F0000 24251 465F4 01670 2E3E0 76C20 74311 5D980 09F20 8F3E0 44BB1 954E0 D7BB1 974E0 ED211 A37AO 5B9E0 O9F20 « CHKR: #2CFA9h
Accrochez les 8 chaînes les unes aux autres en appuyant 7 fois sur la touche +, puis dupliquer la chaîne totale, vérifiez-la avec CHKR, si le résultat est #8E873Ch, tout est ok et vous passer à la suite, sinon, recommencez l'étape V !!!
Il ne vous reste plus maintenant qu'à éxécuter ASSR et stocker le résultat (qui commence par R->B -> V << 0RDZ ...) dans BRDBL.
REMARQUES:
— Le programme BRDBL peut être utilisé comme tout programme ordinaire, mais si vous utilisez la commande VISIT dessus, il faudra terminer votre session par ON, au risque de le détruire.
— Il est bon de ne pas purger ASSR et CHKR car vous n'aurez pas à refaire les étapes I à IV en entrant un autre jeu.

PROGRAMMES POUR HP 48

de Karim ALAF[illegible]



Programme permettant la résolution des équations du second degré (beaucoup plus pratique que HPSOLVE).

```
<< ->ABC
<< 'B^2-*A*C' ->NUM ->
<< « EQ: « A + « X^2+ « + B + « X+ « + C + ' EQDEG2 ' STO
' (-B- V  )/(2*A) ' ->NUM ' X1 ' STO ' (-B+   )/(2*A) ' -> NUM ' X2
'

STO CLLCD « RESOLUTION 2ème DEGRE « 1 DISP 3 FREEZE
' EQDEG2 ' RCL 2 DISP « : « -> STR + 3 DISP « X1 :» X1 -> STR
+4 DISP « X2 :» X2 ->STR + 5 DISP 0 WAIT DROP ' EQDEG2 '
PURGE
>>
>>
>>
```

NOTICE : Entrez les coefficients dans la pile et éxécutez le programme

Calcul du produit vectoriel de deux vecteurs (bi ou tri-dimentionnel). Cette version ne marche pas avec les expressions symboliques.

```
<< RCLMENU 'MZW' STO { { «PROD» } { «VECT» } { «BY»} {
«AKP» }
{ } { «1994» } } MENU «ENTRER VECTEURS» { «:V1:
:V2" { 1 0 } ALG } INPUT OBJ -> OBJ -> DROP OBJ -> DROP
-> X1 Y1 Z1
<< OBJ -> DROP OBJ -> DROP -> X Y Z
<< CLLCD « PRODUIT VECTORIEL:» 4 DISP «    « 'Y*Z1-Z*Y1
EVAL 'Z*X1-X*Z1' EVAL 'X*Y1-Y*X1' EVAL->V3+5 DISP
>>
>> 0 WAIT DROP 'MZW' RCL { CST MZW } PURGE MENU
>>
```

Version avec expressions symboliques.

```
<< -> X Y Z X1 Y1 Z1
<< 'Y*Z1-Z*Y1' EVAL 'X' SWAP = 'Z*X1-X*Z1' EVAL
'Y' SWAP = 'X*Y1-Y*X1' EVAL 'Z' SWAP = >>
```

Notice : Notez les coordonnées (X,Y,Z) des deux vecteurs dans la pile et exécutez.

Suites, programme permettant de trouver les termes d'une suite recurrente (Un+1=f(un)), donnant la variation.

```
<< RCLMENU 'B' STO 'X' PURGE CLLCD { { «SUITES» } { «BY»
} { «AKP» }
{ } { «1993» } { «1994» } } MENU «ENTRER 1er TERME» «:UO:»
INPUT OBJ ->
EVAL «ENTRER FONCTION» { «:F(Un): '« { O O }    ALG } INPUT
«'« +
OBJ-> EVAL O 'A' STO 'F' STO 'UO' STO CLLCD «SUITE:» 1 DISP
«UO=» 'UO'
EVAL + 2 DISP «Un+1=» 'F' EVAL + 3 DISP 3 FREEZE 'UO' EVAL
'X' STO
{ { «Un+1» {
<< «U» 'A+1' EVAL + «=» + 'F' EVAL EVAL + 4 DISP 3 FREEZE 'F'
EVAL
EVAL 'X' STO 'A' ->NUM 1 + 'A' STO
>> } } { «FRAC» { << 'X' EVAL ->Q «U» 'A' EVAL + «=» + SWAP +
4 DISP
3 FREEZE >> } }
{ «VARI» {
<< 'X' EVAL 'XS' STO O 'CXS' STO 'UO' EVAL 'X' STO
O 20
FOR B 'F' EVAL ->NUM DUP 'X' STO
IF 'F' ->NUM <
THEN 'CXS' 1 STO+
ELSE 'CXS' 1 STO-
END
NEXT 'XS' EVAL 'X' STO 'CXS' EVAL ABS
IF 21 ==
THEN 'CXS' EVAL
IF -21 ==
THEN «SUITE DECROISSANTE» 5 DISP 3 FREEZE
ELSE «SUITE CROISSANTE» 5 DISP 3 FREEZE
END
ELSE «SUITE NON MONOTONE» 5 DISP 3 FREEZE
END
>> } }
{ «EXIT» {<< 'B' RCL { A B CXS CST F X UO XS }PURGE MENU
>> } } } MENU
>>
```

NOTICE : Pour des raisons de temps évidentes la variation est testée sur les 20 premiers termes.

Version améliorée de D->R de la calculatrice (fraction)

```
<< ->A << A 180 / EVAL ->Q    * >> >>
```

Notice : Entrez l'angle et exécutez.

Recherche de l'équation de la tangente en un point.

```
<< -> F X0 << F 'X' PURGE X -> F1 <<X0 'X' STO «'Y=»
F1 EVAL  «*X+» + + F EVAL X EVAL F1 * - + «'«+
OBJ->COLCT 'X' PURGE >>
>>
```

Notice : Entrez la fonction et le point, exécutez.

Linéarisation de cos^n et sin^n.

```
<< 0 0 -> N K << << N K COMB N K 2 * - >>
<< IF DUP
THEN 'T' * COS *
ELSE DROP 2 /
END
>> << 2 N 1 - ^ / >>
<<IF2MOD
THEN NEG
END
>> -> P1 P2 P3 P4 << { { «COS^» << ->N << 'COS(T)' N ^ 0 0 N
2 / FLOOR FOR K P1 EVAL P2 EVAL + NEXT P3 EVAL = >> >> }
{ «SIN^» <<-> << 'SIN(T)' N ^0 0 N 2 / FLOOR
IF N 2 MOD
THEN FOR K P1 EVAL 'T' * SIN *K N 1 - 2 / + P4 EVAL + NEXT
ELSE FOR K P1 EVAL P2 EVAL K N 2 / + P4 EVAL + NEXT END P3
EVAL
= >>
>> } { } { } { } { «EXIT» << CONT >> } } TMENU HALT 2 MENU >>
>> >>
```

NOTICE : Entrez le degré et appuyez sur «sin^» ou «cos^».

F O X

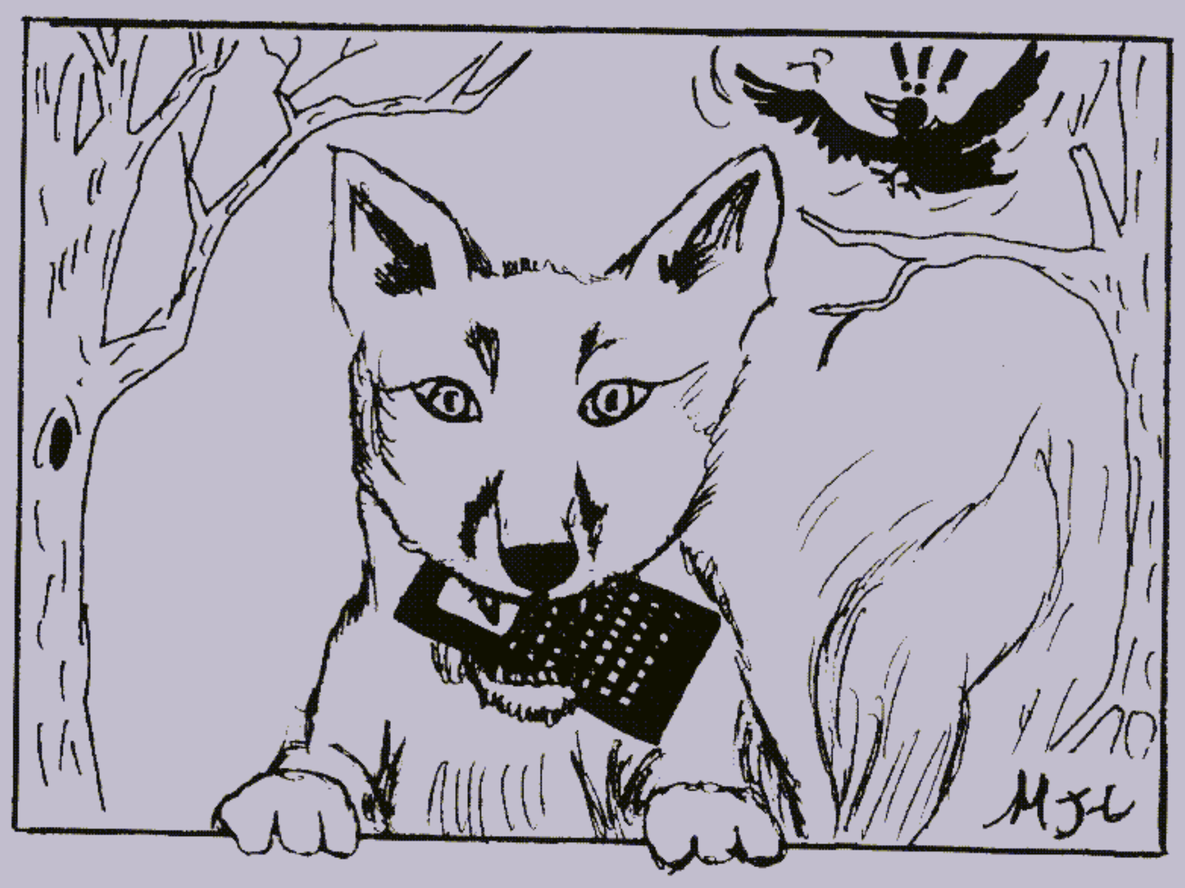
par LAURENS Denis

FOX est un petit jeu d'action au but assez simple.

NOTICE DE JEU

Tout d'abord vous choisissez votre option, c'est à dire joueur débutant ou joueur confirmé. Le choix se fait suivant l'appui sur les touches 1 ou 2.
Sur le bord gauche de l'écran vous est indiqué le niveau de tableau que vous allez accéder, puis un appui sur une touche et vous voila en pleine action. Vous êtes à bord d'un hélicoptère survolant des murs de différentes tailles. Le principe : éviter de les percuter.
Un seul moyen !! Vous devez les détruire à l'aide de bombes placées au dessous de votre hélicoptère que vous larguez par simple pression de la touche +.
Vous disposez de 3 vies, pour accéder au prochain tableau il vous suffit de détruire tous les murs.
Votre hélicoptère descend petit à petit en augmentant sa vitesse.
ATTENTION : Pour stopper le jeu, un seul recours, il faut appuyer sur les touches ON-C.
Remarque : ne jamais appuyer sur la touche ON seule, vous risqueriez d'avoir quelques désagréments à la fin du tableau.
Il est possible d'arrêter le son, il suffit de désarmer le FLAG SON, mais votre hélicoptère aura une vitesse supérieure.
Le jeu fonctionne seulement sur HP48S/SX, l'emploi sur HP48G/GX est possible, il suffira de changer quelques adresses. (BUFFER VOIE SERIE, BITMAT ECRAN .....)
Ce programme assemble les chaines de caractères désignant un objet décompilé autrement dit il le compile.

```
ASS
<< DUP SIZE 2 / .5 + IP «GROB 1 « SWAP + « «
+ SWAP + OBJ # 4017h
SYSEVAL # 56B6h SYSEVAL DROP NEWOB >>

DEBUT ( #274F )
« D9D20
F8040
12040
F2140
F8040
85705
C2A20
13000
13A3445424554514E445020223A334F4E4649425D454
00D11
83405
CCD20
79000
8FB9760  GOSBVL # 0679B          ; SAUVER LES REGISTRES D0, D1, B, D
8F51C10  GOSBVL # 01C15          ; EFFACE LE CONTENU DU GROB MENU
8F7DB10 GOSBVL # 01BD7           ;  ETEINT LES INDICATEURS
1FB0207  D1 (5) # 7020B                  ; CHARGE LE NBRE
3430000          LC (5) # 00003          ; DE VIE POINTE
145              DAT1=C  A               ; A L'ADRESSE D1
80822800 RET LA (3)        # 008
7510             GOSUB TEST
414              GOC DEBUTANT
80822400 LA (3) # 004
7600             GOSUB TEST              ; TEST L'ACTION DES
432              GOC CONFIRME   ; TOUCHES 1 ET 2
51E              GONC RET                ; SI CARRY = 0 ALORS RET, SI=1
CONTINU
8F75D00  TEST GOSBVL # 00D57             ;VIDE LE BUFFER DU CLABIER
32200            LC (3) # 002            ;OUT=C
8FCEE10 GOSBVL 01EEC                     ;«ET LOGIQUE» ENTRE CETTE
                                          REPONSE ET «IN»
0E32             C=C&A X                 ;«ET LOGIQUE» ENTRE CETTE
                                          REPONSE ET «IN»
93E              ? C # 0 X               ;LA TOUCHE EST-ELLE ENFONCEE
00               RTNYES                  ;RETOUR DE SOUS PROGRAMME,
                                          AVEC BIT CARRY ARME
01               RTN                     ;RETOUR AVEC BIT CARRY
                                          DESARME
39FFF001000      CONFIRME LC (A) # 0000100FFF      ;
4EO      GOC SORT                        ;C=VITESSE, DIFFICULTE
39FFF0030000 DEBUTANT LC (A) # 0000300FFF          ;
1F10207          SORT D1 (5) # 70201     ;LE REGISTRE C
15D8             DAT1=C  9               ;EST CHARGE A L'ADRESSE
POINTEE PAR D1
8D34150  GOVLNG # 05143                  ;RESTAURE LES REGISTRES D0, D1,
B, D ET EXIT
B2130    «
FOX ( # 2155 )
« D9D20
```




```
76BA1
3FC11
83405
56F14
FEF30
FEF30
250F4
BECF1
CCD20
25300
8FB9760  GOSBVL # 0679B
8FFFB10 GOSBVL # 01BFF                   ;EFFACE LE GROB ECRAN
30F              LC (1) # F              ;
8082102          LA (2) # 20             ;
1BE5917  DO (5) # 7195E                  ;DO POINTE SUR LA 64 LIGNE DE
                                          LA BITMAP ECRAN
1540 LIGNE DATO=C        P               ;CHARGE LE REGISTRE C A
                                          L'ADRESSE DO
160              DO=DO+1                 ;
A6C              A=A-1    B              ;ON REPETE 21 FOIS LA SEQUENCE
55F              GONC LIGNE              ;RETOUR SI CARRY=O
```

FOX pour HP48

```
D1                  B=O      A
34C3917  LC (5) # 7193C                    ;
06       BAT1       RSTK=C                 ;
134                 DO=C                   ;
1F63100             D1 (5) # 00136         ;D1 POINTE SUR L'HORLOGE
147                 C=DAT1 A               ;C = RAND (F)
20       BAT2       P=O
80825FF18FF         BAT3     LA (6) # FF81FF   ;
148                 DATO=A B                   ;LE PROGRAMME BAT3
18F                          DO=DO-10          ;AFFICHE UN GROB 2LI * 3COL
18F                          DO=DO-10          ;A LA 63 LIGNE DE LA BITMAP
                                                ECRAN
181                 DO=DO-2                    ;CE GROB REPRESENTE UNE CASE
                                                DU MUR
E5                  B=B+1    A                 ;B = NBR DE COL
OD                  P=P-1                      ;
OD                  P=P-1                      ;
88A                 ? P # A                    ;TANT QUE P # A ON RECOM
                                                MENCE
OE                  GOYE     BAT3              ;
A2E                 C=C-1    XS                ;DEC C, ON EMPILE C * FOIS DE
                                                CASE
58D                 GONC     BAT2              ;AFIN DE CONSTITUER LE MUR,
                                                TESTE CARRY
07                  C=RSTK
818F22              C=C+3    A                 ;INC C DE 3
20                  P=O                        ;
80821D5  LA (2) # 5D                           .CONTROLE LA POSITION DES
                                                MURS
9E6                          ? A>C    B        ;SI ON SE TROUVE DANS PICT ON
                                                CONSTRUIT
3B                  GOYE     BAT1              ;UN AUTRE MUR 3 LIGNES PLUS
                                                LOIN
1BCF107 DO (5) # 701FC                         ;DO POINTE SUR LE BUFFER DU
                                                PORT SERIE
D4                  A=B      A                 ;AU NBR DE COL DE TOUT LE
                                                GROB
140                          DATO=A A          ;CHARGE LE REGISTRE A A
                                                L'ADRESSE POINTE PAR DO
3400117             LC (5) # 71100             ;DO POINTE SUR LE DEBUT DE LA
                                                BITMAP ECRAN
8082E100C13006250054         RETa     LA (F) # 45005260031C001    ;
7240                GOSUB    DESSIN            ;
881                 ? P # 1                    ;
5E                  GOYE     RETa              ;
20                  P=O                        ;
8082EDFFDF10008FFFFF         RETb     LA (F) ;# FFFFF80001FDFFD          ;
7320                GOSUB    DESSIN            ;
881                 ? P;# 1                    ;
5E                  GOYE     RETb              ;
20                  P=O                        ;LES PROGRAMMES RETa, RETb,
                                                RETc
80829080200EFFO      RETc     LA (A) # 0FFE002080        ;
7900                GOSUB DESSIN               ; AFFICHENT L'HELICOPTERE SUR
                                                L'ECRAN
886                 ? P # 6                    ;ILS CONSTITUENT UN GROB 5LIG
                                                * 8 COL
AE                  GOYE     RETc              ;LE SOUS PROGRAMME DESSIN
6C10                GOTO CONT                  ;CHARGE LES VALEURS SUCCESSI
                                                VES
1584 DESSIN         DATO=A 5                   ;DU REGISTRE A
16F                 DO=DO+10                   ;A L'ADRESSE POINTEE PAR DO
16F                 DO=DO+10                   ;SUR 5 QUARQUETS
161                 DO=DO+2                    ;IL REALISE AINSI UN AFFICHAGE
OD                  P=P-1                      ;LIGNE PAR LIGNE
OD                  P=P-1                      ;SUIVANT LES VALEURS DE P
OD                  P=P-1                      ;LE REGISTRE A SE COMPRTE
OD                  P=P-1                      ;COMME UN REGISTRE A
DECALAGE
OD                  P=P-1                      ;RETa GROB 5LI * 3COL, RETb 5LI *
3COL
01                  RTN                        ;RETc 5LI * 2COL, RETOUR DE
                                                SOUS PROG
06       CONT       RSTK=C                     ;SAUVE LA VALEUR COURANTE DE
                                                L'AFFICHAGE
20       P=O
3401000             LC (5) # 00010             ;
D7                  D=C      A                 ;
DB       MUS        C=D      A                 ;
07                  C=RSTK                     ;EST UNE ROUTINE DE SON
D7                  D=C      A                 ;IL ESSAI AU MAXIMUM DE
                                                REPRODUIRE
3402000             LC (5) # 00020             ;LE BRUIT D'UN HELICOPTERE

C3                  D=D+C    A                 ;LES 4 VALEURS SUCCESSIVES
3400100             LC (5) # 00100             ;DU REGISTRE C REPRESENTENT
8BF                 ? D<=C   A                 ;UNE SEQUENCE DE DEPART, DE
FIN
7D                           GOYE     MUS      ;UN INCREMENT DE FREQUENCE,
                                                UNE DUREE DE PALIER
8F75D00  GOSBVL # 00D57              ;         VIDE LE BUFFER DU CLAVIER
32100               LC (3) # 001               ;CODE DE LA TOUCHE +
80822100 LA (3) # 001                ;
801                 OUT=C                      ;
8F06110             GOSBVL # 01160             ;
0E32                C=C&A X                    ;ET LOGIQUE
93E                 ? C=O    X                 ;SI LA TOUCHE EST ENFONCEE
CARRY=1
14                  GOYE     TIR               ;CARRY=1 ALORS ROUTINE DE TIR
07IC1               C=RSTK                     ;RECUPERATION DE LA
                                                VITESSE COURANTE DE C
1F10207             D1 (5) # 70201             ;D1 POINTE SUR L'ADRESSE
                                                VITESSE
143                 A=DAT1   A                 ;A=VALEUR VITESSE
CC BOUCL            A=A-1    A                 ;C'EST LA BOUCLE D'ATTENTE
5DF                 GONC     BOUCL             ;DE LA VALEUR VITESSE
134                 DO=C                       ;DO POINTE SUR LA VALEUR
                                                COURANTE
DO                  A=O      A                 ;
27                  P=7                        ;
1580     EFF        DATO=A 1                   ;CHARGE LE REGISTRE A LA
                                                VALEUR COURANTE
16F                 DO=DO+10                   ;DE L'ECRAN, DO EST INC DE 22
16F                 DO=DO+10                   ;DU FAIT DE LA SYNCHRONISA
                                                TION DE
161                 DO=DO+2                    ;L'IMAGE EN COURS, ON EFFACE
                                                AINSI
OD
                    P=P-1
                                                ;1LI * 8COL DU GROB DE L'HELI
                                                COPTERE
50F                 GONC     EFF               ;
161                 DO=DO+2
142                 A=DATO 2                   ;CONTROLE SI L'HELICOPTERE EST
92C                 ? A # O  XS                ;RENTRE EN COLLISION ARME
                                                CARRY
80                  GOYE     TOUCHE            ;SI CARRY=1, ROUTINE TOUCHE
E6                  C=C+1    A                 ;INC DE 1 LA VALEUR COURANTE
61FE                GOTO     REV               ;L'HELICOPTERE AVANCE D'UNE
                                                POSITION
6BF0 TOUCHE         GOTO CRASH
07       TIR        C=RSTK                     ;SAUVE LA VALEUR COURANTE
D5                  B=C      A
06                  RSTK=C
80824CCOOO          LA (5) # OOOCC             ;
C8                  B=B+A    A                 ;ON PLACE LA BOMBE
D9                  C=B      A                 ;SOUS L'HELICOPTERE
134                 DO=C
20 REPART           P=O
80821FF  BOMB LA (2) # FF                      ;GROB DE LA BOMBE 1LI * 1COL
1580                DATO=A 1                   ;ON AFFICHE LA BOMBE
16F                 DO=DO+10                   ;  \
16F                 DO=DO+10                   ;  > SYNCHRONISATION
161                 DO=DO+2                    ;  /
OD                  P=P-1                      ;
88E                 ? P # E                    ;
7E                  GOYE     BOMB              ;TEST LA CARRY
8082400003          LA (5) # 30000             ;C'EST LA
CC BOMBOUCL         A=A-1    A                 ;D'ATTENTE
5DF                 GONC     BOMBOUCL
134                 DO=C                       ;DO POINTE SUR LE DEBUT DE LA
                                                BOMBE
DO                  A=O      A                 ;A=OOOOO
22                  P=2                        ;
1580 BOMBEF         DATO=A 1                   ;ON AFFICHE UN GROB VIDE
16F                 DO=DO+10                   ;A LA POSITION ACTUELLE DE
16F                 DO=DO+10                   ;LA BOMBE, ON L'EFFACE
161                 DO=DO+2                    ;
OD                  P=P-1                      ;
50F                 GONC     BOMBEF            ;
20                  P=O
8082422000          LA (5) # 00022             ;
C2                  C=C+A    A                 ;INC DE LA POSITION DE LA
```

FOX pour HP48

```
51 if mode$<>»curseur» then 53
  if pp=1 then gosub 45
  pset(x,y),pixel xor indpt
9B2                 ? C>A     X
FO                  GOYE      FINO
1520                A=DATO P
90C                 ? A # O   P
AO                  GOYE      IMM
639F                GOTO      REPART
6C3F                FINO      ICI
1F10207  IMM        D1 (5) # 70201
143                 A=DAT1  A
CC                  A=A-1     A
141                 DAT1=A  A
1F60207             D1 (5) # 70206
147                 C=DAT1  A
80DO                P=C       O
DO                  A=O       A
180                 DO=DO-1
OD                  P=P-1
1503 EFFMUR         DATO=A X
16F                 DO=DO+10
16F                 DO=DO+10
161                 DO=DO+2
OD                  P=P-1
50F                 GONC      EFFMUR
20                  P=O
1520                A=DATO P
908                 ? A=O     p
42                  GOYE      DECDO
80821FF EGADO LA (2) # FF
148                 DATO=A B
1FCF107  D1 (5) # 701FC
143                 A=DAT1  A
EA                  A=A-C     A
8A8
                    ? A=O
                              A
FO                  GOYE      FINI
141                 DAT1=A  A
6C9F                GOTO      FINO
160 DECDO           DO=DO+1
5AD                 GONC      EGADO
8D34150 FINI        GOVLNG # 05143
E6 CRASH            C=C+1     A
06                  RSTK=C
20                  P=O
368088001           LC (7) # 1008808
8082101             LA (2) # 10
1F01207             D1 (5) # 70210
15D6                DAT1=C  7
07 CRAC C=RSTK
134                 DO=C
06                  RSTK=C
27                  P=7
15E3 PEPE           C=DATO 4
8F01207             GOSBVL # 70210
15C3                DATO=C 4
16F                 DO=DO+10
16F                 DO=DO+10
161                 DO=DO+2
```

```
;LA BOMBE EST-ELLE EN FIN DE
 COURSE ?
;SI OUI BRANCHEMENT SI NON
 CONTINUE
;A T'ON
;FAIT MOUCHE ?
;
;LA BOMBE DESCEND

;D1 POINTE SUR LA VALEUR
 VITESSE
;A = VITESSE
;VITESSE = VITESSE - 1
;CHARGE LA NOUVELLE VALEUR
;DANS D1
;D1 POINTE SUR LE CHOIX DE
 DIFFICULTE
;C = CHOIX
;P = AU PREMIER QUARTET DE C

;DO POINTE SUR LA POSITION
 D'UNE CASE DU MUR
;C'EST LA ROUTINE D'EFFACE
MENT
;DE LA CASE DU MUR SUIVANT
;LA VALEUR DE P REPRESENTANT
 LE CHOIX
;DEC CHOIX
;

;REAJUSTE LA BARRE DU MUR A
 AFFICHER
;GROB DE LA BARRE DU MUR 2LI *
 1COL
;ON L'AFFICHE
;D1 POINTE SUR LE NBRE DE
 CASES EMPILEES
;A = NBRE EMPILE
;SOUSTRAIT LE NBRE DE CASES -
 LE CHOIX

;A-T'ON DETRUIT TOUS LES MURS
;SI OUI RETOUR RPL
;SINON CHARGE LA NOUVELLE
 VALEUR A
;
;CALCULE POSITION BARRE DU
 MUR
;PUIS ON L'AFFICHE
;RESTAURE DO, D1, B, D ET
 RETOUR RPL

;LE REGISTRE C REPRESENTE UN
;SOUS-PRG
;CBIT=O O, RTN
;D1 POINTE SUR LE BUFFER SERIE
;ON CHARGE LE SOUS-PRG

;DO = VALEUR COURANTE

;C=GROB HELICO 5LI * 1COL
;APPEL DU SOUS-PRG HYBRIDE
;ON RECHARGE LE GROB
 HELICOPTERE
;MAIS AVEC LE A-1 BIT DE C=O
;CE PROCEDE PERMET UNE
;DESTRUCTION
;PAS A PAS DE L'HELICOPTERE
```

```
20                  P=O
34FFFFF L (5) # FFFFF
A3E LCUOB           C=C-1     X
5CF                 GONC      LCUOB
1D41                D1 (2) # 14
BIT C
A6C                 A=A-1     B
1590                DAT1=A  1
CBIT O (A)
96C                 ? A # O   B
DB                  GOYE      CRAC
1FB0207  D1 (5) # 7020B
147                 C=DAT1  A
CE                  C=C-1     A
8AA                 ? C=O     A
12                  GOYE      PERDU
145                 DAT1=C  A
8F75D00  GOSBVL # 00D57
8F99940 COP         GOSBVL # 04999
58F                 GONC      COP
163                 DO=DO+4
07                  C=RSTK
CE                  C=C-1     A
06                  RSTK=C
6BFE                GOTO      IMM
8F2D760 PERDU GOSBVL # 067D2
D0                  A=O       A
8DC7530 GOVLNG # 0357C
B2130     «
JEU       ( # C7D5 )
<<DEBUT 1 100 FOR I «NIVEAU : « I
FOR IF DEPTH 1 = THEN CLEAR
«PERDU ..... NIVEAU :» I + DOERR END
TEXT NEXT >>
D'OU LES CODES A ENTRER :
DEBUT (# 274F )
«D9D2OF8O4O12O4OF214OF8O4O857O5C2A2O
13A3445424554514E445O2O223A334F4E464
9425D454OOD11834O5CCD2O79OOO8FB976O8
F51C1O8F7DB1O1FBO2O7343OOOO1458O8228
OO751O4148O8224OO76OO43251E8F75DOO32
2OO8FCEE1OOE3293EOOO139FFFOO1OOOO
4EO39FFFOO3OOOO1F1O2O715D88D3415OB213O»

FOX ( # 2155 )
«D9D2O76BA13FC11834O5   56F14FEF3OFEF3O
25OF4BECF1CCD2O253OO   8FB976O8FFFB1O3O
F8O821O21BE5917154O16OA6C55FD134C391
7O61341F631OO1472O8O825FF18FF14818F1
8F181E5ODOD88AOEA2E58DO7818F222O8O82
1D59E63B1BCF1O7D414O34OO1171342O8O82
E1OOC13OO625OO54724O8815E2O8O82EDFFD
F1OOO8FFFFF732O8815E2O8O829O8O2OOEFF
O79OO886AE6C1O158416F16F161ODODODODO
DO1O62O34O1OOOD7DBO6345OOOO8F6A71OO7
D734O2OOOC334OO1OO8BF7D8F75DOO321OO8
O8221OO8O18FO611OOE3293E14O71F1O2O71
43CC5DF134DO27158O16F16F161OD5OF1611
4292C8OE661FE6BFOO7D5O68O824CCOOOC8D
91342O8O821158O16F16F161OD88E7E8O824
OOOO3CC5DF134DO22158O16F16F161OD5OF2
O8O82422OOOC28O8229199B2FO152O9OCAO6
39F6C3F1F1O2O7143CC1411F6O2O71478ODO
DO18OOD15O316F16F161OD5OF2O152O9O842
8O821FF1481FCF1O7143EA8A8FO1416C9F16
O5AD8D3415OE6O62O368O88OO18O821O11FO
12O715D607134O62715E38FO12O715C316F1
6F161OD55E2O34FFFFFA3E5CFD41A6C159O9
6CDB1FBO2O7147CE8AA121458F75DOO8F999
4O58F163O7CEO66BFE8F2D76ODO8DC753OB213O»
```

```
;
;C'EST LA BOUCLE D'ATTENTE
;
;ON POINTE SUR LA POSITION DU

;DU SOUS-PRG, PUIS ON DEC DE 1
;CHARGE NOUVELLE VALEUR DE

;L'HELICOPTERE EST-IL
DETRUIT ENTIEREMENT ?
;CARRY=1 DETRUIT
;D1 POINTE SUR VALEUR VIE
;C=VIE
;DEC VIE DE 1
;VIE=O ?
;SI OUI PERDU
;SI NON CHARGE NOUVELLE
VALEUR VIE

;ATTEND L'APPUIU D'UNE TOUCHE
;TOUCHE NON ACTIONNEE ON
 RECOMMENCE

;RECUPERE DO, D1, B, D

PLACE SUR LA PILE L'ENTIER
 SYSTEME A
```

RENTREZ LES CHAINES SANS ESPACE ET SANS SAUT DE LIGNE.
UNE FOIS LES CHAINES TAPEES, ENTEZ DUP BYTES
ET VERIFIEZ BIEN QUE LES CHECKSUM DES PROGRAMMES CORRESPON-DANTS
SI OUI LES COMPILER AVEC LE PROGRAMME ASS.
UNE FOIS FAIT, TAPEZ LE PROGRAMME JEU, ET LANCEZ SON EXECUTION.

RUBRIQUE CASIO FX 850/880P

Transformez votre casio FX 850 en 880 en 10 étapes...

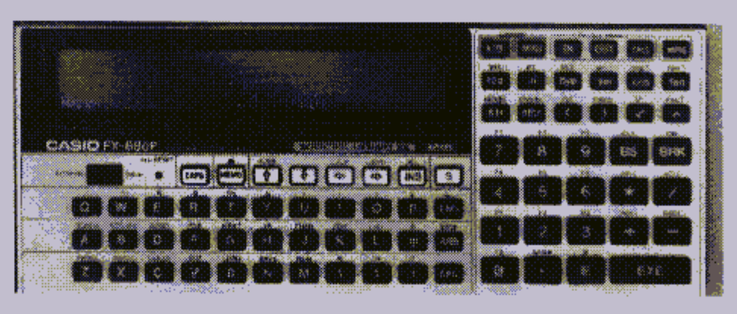
POCHE COMPUTERS et CASIO News vous proposent aujourd'hui de transformer votre FX-850P en FX-880P, donc de l'étendre de 32 Ko, en seulement 10 étapes. Mais cette transformation nécessite une habileté et un matériel particulier, alors regardez attentivement les recommandations avant de vous lancer dans cette aventure... Enfin, nous ne pourront être en aucun cas tenus pour responsables des éventuelles pannes survenues à la suite de la réalisation de cette opération. Voilà tout est dit alors c'est parti...

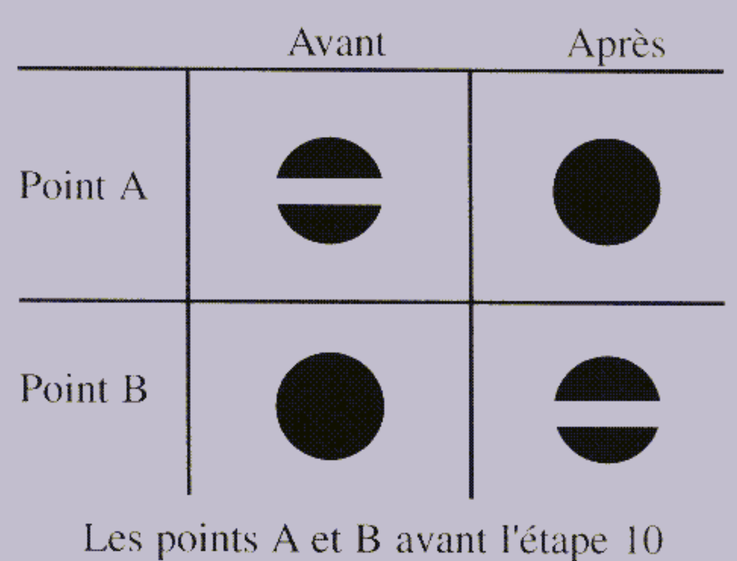
Matériel indispensable :
- Fer à souder de puissance mini 15 W
- Soudure étain de diamètre <= 0,8 mm
- Tournevis cruciforme de faible diamètre
- SRAM 43256, 62256, 58256 ou compatible 32K octets (150 ns maximum).

Conseils
- utilisez un aimant pour éviter de perdre les vis.
- utilisez une boite pour mettre les différentes pièces.
- utilisez une éponge pour nettoyer le fer au fur et à mesure que le travail avance.

10 étapes...

1- Démonter le boîtier en laissant la plaque noire.
2- enlever les 2 piles CR2032 et la CR1220,
3- dévisser et enlever les vis de la plaque noire.
4- déplier la plaque noire vers le haut (Attention aux fils du buzzer).
5- extraire les 2 ressorts de couleur or.
6- Maintenant vous entrez dans la partie la plus délicate de l'opération alors si vous ne vous sentez pas vraiment à la hauteur : **STOOOOOOOOP**, vous avez la vie de votre machine dans vos mains... Pour les autres, dessoudez le circuit SRAM 8 Ko le plus à gauche et le plus près du connecteur pour RP33 (il est souvent noté HM6264A).
7- dessouder le point noté B.
8- souder le nouveau circuit SRAM 32 Ko à la place de l'ancien.
9- souder le point noté A.
10- remettre les ressorts, la plaque noire, les vis, les piles...





Les points A et B avant l'étape 10

Les tests

Voilà, il ne reste plus qu'à effectuer les tests d'usage :
- Allumer votre machine,
- faire un reset complet (ALL RESET + P derrière la machine),
- puis 0400 + LIB.

Si la machine indique MAIN 32 Ko, vous avez réussi, BRAVO... Pour toutes questions complémentaires concernant cette extension, écrivez en BAL TonyCrook sur le serveur Eurêka (99.36.48.43.).

CASIO News

serveur Eurêka
au 99.36.48.43
bal TonyCrook

ou à l'adresse suivante :

CASIO News
Anthony TARDIVEL
La Gravelle
35410 NOUVOITOU

EDITEUR DE MEMOIRE (par Christophe BULLA)

Ce programme édite sur l'écran le contenu de la mémoire, mais aussi les adresses qui peuvent être utiles.

```
10 CLS: P=0: PRINT CHR$(9) "*** OISAC 2 *** EDITEUR-MEM ***ADRESSE        PAGE            :";:
LOCATE 10,1: INPUT@7;A: LOCATE0,0: LOCATE 25,1/ INPUT@3;P: DEFSEG=P*4096: Z=0: CLS
20 H$="": A$="": FOR I=0TO14: D=PEEK(A+I): HS=HS+RIGHTS(HEX$(D),2): IF D<32 THEN D=46
30 A$=A$+CHR$(D):NEXT
50 CLS: IF Z THEN 110
100 PRINT"  "HS"  &H"HEXS(A)"  &H" RIGHTS(HEX$(P),2)"   ["AS"] :: GOTO 120
110 PRINT"   "H$"    &H"MID$(HS,3,2)LEFTS(H$,2)"&H"  &H"MID$(HS,5,2)"  &H"MIDS(H$,9,2)
MID$(HS,7,2)"   [MID$(AS,8,8)"]":
120 $=INPUT$(1,@):CLS
130 IF $="1"OR$="0"THEN Z=VAL($): GOTO 50
135 IF $="*" THEN 10
140 IF $="+" THEN A=A+15: GOTO 20
145 IF $="-" THEN A=A-15: GOTO 20
150 IF S="/" THEN INPUT "DECALAGE :   ",Y: A=A+Y: GOTO 20
155 IF S<>"H" THEN 160 ELSE PRINT"Hexa -> Deci :  ":: INPUT@6;Y: IF Y<0 THEN Y=65536+Y
156 PRINT Y"d": GOTO 50
160 IF $="D" THEN PRINT"Deci -> Hexa :  ":: INPUT@6;Y: PRINT"&H"HEXS(Y): GOTO 50
170 BEEP: GOTO 50
```

La première chose que demande le programme est l'adresse où vous désirez aller, et la page de cette adresse. Ci-dessous voici la liste des touches à utiliser et les 2 types d'écran affichés.
[1] Passe en mode DUMP
[0] Passe en mode adresse
[*] Revient au début du prg pour changer l'adresse et de page
[/] Pour effectuer un décalage de l'adresse en cours
[+] Avance de 15 octets
[-] Recule de 15 octets
[H] Permet de faire une conversion Héxadécimale --> Décimale
[D] Permet de faire une conversion Décimale --> Héxadécimale

RUBRIQUE CASIO FX 850/880P

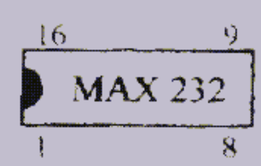
Interface RS232C pour FX-850/880P...

Poche Computers et CASIO News vous proposent ce mois-ci de continuer l'article du second numéro (Trucs et astuces - Page 43), par la parution du schéma pratique de cette interface.

La réalisation de celle-ci est relativement simple et ne devrait donc poser aucun problème. Nous allons cependant récapituler quelques points essentiels puis nous terminerons cet article par le schéma pratique.

Le MAX232

Le MAX232, comme tous les circuits intégrés, est organisé de la façon suivante : Nous conseillons grandement l'utilisation d'un support de CI pour ce composant.



Les connecteurs

	CASIO FX-850/880P	PC/ST/AMIGA 9 broches*	PC/ST/AMIGA 25 broches*
TXD	22	3	2
RXD	5	2	3
RTS	7	7	4
CTS	17	8	5
DTR	21	4	20
DSR	16	6	6
GND	30	5	7

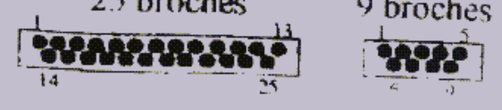
* Sur PC/ST/AMIGA, relier la RTS avec la CTS et la DTR avec la DSR.

Nomenclature

REG = Régulateur 7805
C6 = Condensateur 100nF > 16V
C1-C5 = Condensateur 22uF 16V
R1-R4 = Résistances 100K
R5 = Résistance 1K

Connecteur DB30 CASIO

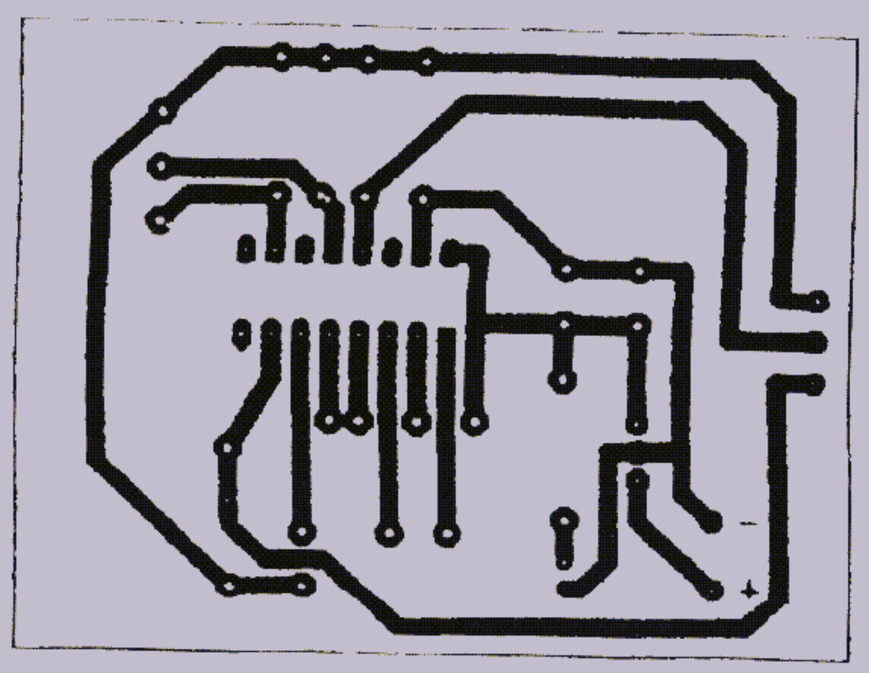
Connecteur PC/ST/AMIGA
25 broches 9 broches

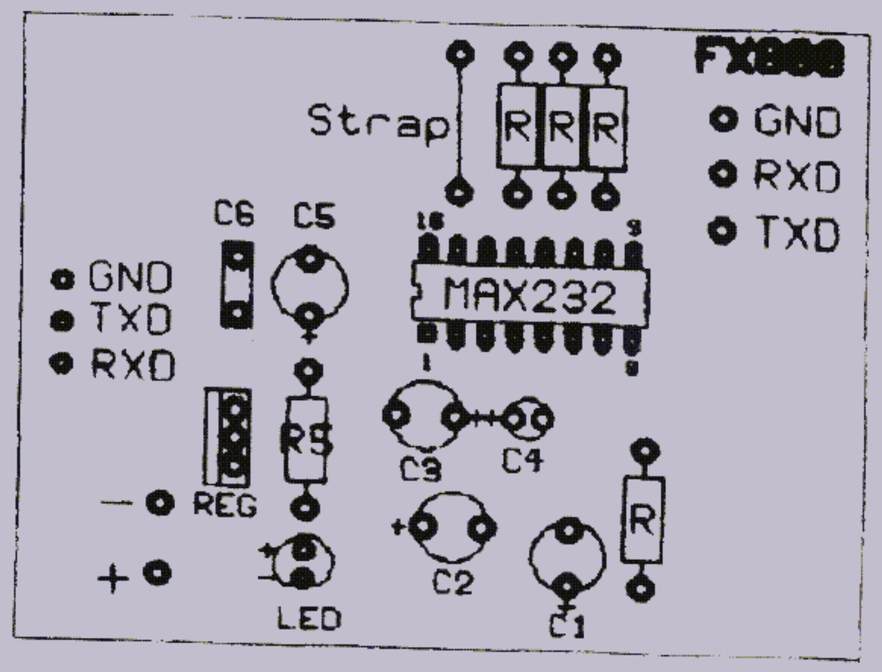


Plus...

Support de CI, support de pile 9V, connecteur 9 ou 25 broches et LED. Pour le connecteur 30 broches de la CASIO FX-850/880P, vous pouvez contacter la société JAPEL à Paris et le commander comme pièce de service après-vente. Il vous en coutera environ 80 francs TTC (frais de port compris).

Schéma pratique





Pour toutes questions complémentaires, écrivez en BAL TonyCrook sur le serveur Eurêka (99.36.48.43.) ou à l'adresse suivante : CASIO News - Anthony TARDIVEL - La Gravelle - 35410 NOUVOITOU.

PROGRAMMES POUR CASIO FX850/880P

par Philippe GOURVES

LECTURE DE LA MEMOIRE:

C'est un programme tout simple et que beaucoup doivent avoir sous diverses formes. Le programme demande la valeur de départ de la lecture, puis il affiche sur la première ligne l'adresse, la valeur décimale de l'adresse et le caractères ASCII correspondant.

La deuxième ligne affiche en continu sur les 32 caractères ASCII correspondant aux valeurs successives de la mémoire.

Le programme fonctionne en continu mais il est facile de le faire fonctionner adresse par adresse en supprimant le point-virgule en fin de ligne 80 par exemple.

LECTURE DE LA MEMOIRE. (215 octets)

```
10 CLS:PRINT "D=";D;:INPUT D
20 D1=INT(D/256)*256:D2=D-D1
30 DEFSEG=D1/16
40 FOR A=D2 TO 65536
50 XP=PEEKA:IF XP(32 THEN X=32 ELSE X=XP
60 N=N+1:IF N>29 THEN N=0
70 LOCATE0,0:PRINT A+D1;XP;CHR$(X);"          ";
80 LOCATE N,1:PRINT CHR$(X);">";
100 NEXT
```

LONGUEUR D'UN PROGRAMME.

Ce programme détermine l'adresse de début, de fin, et la longueur d'un programme No 0 à 9 ou de la mémoire (no 10) ; le programme est basé sur le principe suivant:

Les adresses utiles à l'ordinateur sont stokées en mémoire sur 3 octets successifs. Si l'octet de base est x, on peut déchiffrer ces adresses grâce à cette ligne:

PEEK(x)+PEEK(x+1)*256+PEEK(x+2)*65536

Ainsi, si on désire connaître l'adresse de fin de la RAM (AFR, qui correspond à la quantité de RAM de votre ordinateur), il suffit de remplacer dans la ligne ci-dessus x par 1715.

De même, toutes les adresses de début et de fin de programme sont stockées dans une zone située entre 165 et 11 octets avant l'AFR suivant le tableau ci-dessous:

	x début	x fin
mémo	AFR-165	AFR-162
P9	AFR-150	AFR-147
P8	AFR-135	AFR-132
P7	AFR-120	AFR-117
...		
P0	AFR-15	AFR-12

Pour utiliser ce tableau, déterminer d'abord AFR. Déterminer ensuite x début et x fin dans le tableau, puis réutiliser la ligne de commande ci-dessus pour avoir l'adresse du début ou de fin du programme. Le programme fait tout cela automatiquement.

LONGUEUR D'UN PROGRAMME. (336 octets)

```
10 CLEAR:CLS
12 ADB=PEEK(1715)+PEEK(1716)*256-165
15 LOCATE 0,0:INPUT "Long programme No";N
20 DEFSEG=0:LOCATE 0,1:PRINT "  ";
30 DP=PEEK(ADB+(10-N)*15)+PEEK(ADB+1+(10-
   N)*15)*256+PEEK(ADB+2+(10-N)*15)*65536
40 FP=PEEK(ADB+3+(10-N)*15)+PEEK(ADB+4+(10
  -N)*15)*256+PEEK(A
   DB+5+10(10-N)*15)*65536
50 L=FP-DP:LOCATE 0,1:PRINT DP;" ->";FP;"
 L=";L; :GOTO 15
```

VISUALISATION DE LA LIBRAIRIE

Ce programme utilise le même principe que le précédent. Les programmes de la librairie sont stockés en ROM de la même façon que les programmes de la RAM. Il y a une zone en ROM qui indique les adresses de début et de fin de chaque programme de librairie. Il suffit donc d'aller lire ces adresses dans la ROM et de les substituer à celles d'un programme de la RAM, PO par exemple. L'ordinateur ira lire PO dans la ROM sans toucher aux programmes de la RAM. J'ai rajouté à ce programme une partie permettant de restaurer automatiquement les anciennes valeurs de début et de fin de PO, ce programme lorsqu'on a fini de visualiser la librairie.

Remarques:

Ne pas tenter de modifier un programme de la librairie situé en PO : Il est en mémoire morte et resterait donc tel quel. Par contre l'ordinateur risque de modifier les adresses de début et de fin des autres programmes, ce qui se termine en général par un «all reset» pour cause de perte des repères de début et de fin des programmes dans la RAM.

De même, ne pas tenter d'effacer (new) une librairie en PO. Voir ci-dessus: mêmes causes, mêmes effets.

Par contre, on peut imprimer et sauvegarder toute librairie se trouvant en PO.

Ne pas charger ce programme en PO, sinon attention au piège lorsque la librairie sera elle aussi en PO.

VISUALISATION DE LA LIBRAIRIE. (970 octets)

```
5 CLEAR:CLS:DEFSEG=0:GOSUB200
10 INPUT»Fichier 'lib' numero «;N$:IF VAL(N$)=0
   THEN CLS:GOTO10
20 A=PEEK1715+PEEK1716*256-15
40 D-PEEK1718+PEEK1719*256+PEEK1726*65536
50 F=PEEK1724+PEEK1725*256+PEEK1726*65536
60 FOR J=D TO F-15 STEP15:DEFSEG=J/16:O=J
   -INT(J/16)*16+6
 70 FOR K=1 TO LEN(N$):IF
    ASC(MID$(N$,K))=PEEK(K+O) THEN NEXT:GOTO120
80 NEXT:CLS:GOTO10
120 FOR K=0 TO  5:DEFSEG=J16:B=PEEK(K+JINT(J:16)*16):
   DEFSEG =0:POKEA+K,B:NEXT
130 PRINT»OK P0 : RUN P0 ?»:GOTO#0
200 ' EFFACE P0
 205 ADB=PEEK1715+PEEK1716*256-165
 210DP0=PEEK(ADB+150)+PEEK(ADB+151)*256+PEEK(ADB+
      152)*65536:FP0=PEEK(ADB+153)+PEEK(ADB+154)*256
      PEEK(ADB+155)*65536:DP1=PEEK(ADB+135)+PEEK
      (ADB+136)*256+PEEK(ADB+137)*65536
220 IF FP0=DP1 AND FP0-DP0=1 THEN RETURN
230 IF FP0<16219 AND FP0>2945+FRE2 THEN PRINT
    »Programme dans P0 !»
240 IF FP0<16218 THEN RETURN
245 PRINT»Fichier 'lib' dans P0 !»
 250 PRINT»Suppression fichier ?»:POKE(ADB+153),
     PEEK(ADB+135)
 260 POKE(ADB+154),PEEK(ADB+136)
 270 DP0=2944+FRE2:DPB=INT(DP0/256):DPA=DP0-DPB*256
 280 POKE(ADB+150),DPA:POKE(ADB+155),0
 290 POKE(ADB+152),0:POKE(ADB+155),0
 300 LOCATE25,1:PRINT»OK !»;TAB(0);:GOTO 200
```

Démo graphique pour Casio 850/880P
(par Karl GALLOPIN)

"Creepers"

```
10 Cls : Clear : Defchr$(252)="B9557555B9":Defchr$(253)="3945554539":
         Defchr$(254)="397D6D7D39"
20 A=0:B=31:C=1
30 For X=AtoB StepC
40 A$=Inkey$
50 If A$="A"and Y<>1 then Y=Y+1
60 IfA$="-" and Y <>0 then Y=Y-1
70 Cls: Locate J,K: Print chr$(254);: Locate H,I: Print chr$(253);: LocateF,G:
         Print chr$(254);:Locate D,E: Print chr$(253);: Locate X,Y:
         Print chr$(252);: J=H: K=I: H=F: I=G: F=D: G=E: D=X: E=Y
80 Next X
90 If A=31 then 20 else A=31: B=0:C=-1: Goto 30
```

La touche "A" dirige la chenille vers le bas de l'écran ; la touche "-" la dirige vers le haut.

Dans le prochain numéro de POCHE COMPUTERS, nous publierons une étude en plusieurs parties nommée "Voyage au centre de la FX850P", ou la doc technique complète de cette calculatrice... FABULEUX !!!

TRUCS ET ASTUCES

Truc pour Casio FX 880P : (Bertrand PONCIN)

Pour supprimer un pas : POKE 1867,0

Edition d'un programme sur TI-82 (Loïc FIEUX)

Pour éditer un programme déjà saisi on utilise normalement l'option EDIT du menu programmation accessible par [PRGM]. Dans certains cas, il peut être intéressant de ne pas passer par l'option EDIT, surtout lorsque la liste des programmes disponibles est très longue et que le programme à éditer est au bout de cette liste... On peut alors utiliser l'option NEW afin d'éditer (et non de créer) un programme. Il suffit d'introduire un nom de programme déjà existant lorsque la TI-82 vous demandera de saisir un nom. Le programme sera alors édité.

Saisie d'une fonction sur Sharp PC1403 (Loïc FIEUX)

Voici pour nous une nouvelle occasion d'apprécier l'ingéniosité des utilisateurs de PC1403 d'outre-Rhin, en effet, l'astuce qui suit nous a été communiquée par un club allemand. Elle permet de saisir une fonction pendant l'exécution d'un programme. Il ne sera donc plus nécessaire d'éditer un programme afin de modifier une fonction (par exemple SQR(4*X^2)) utilisée par celui-ci. Désormais vous n'avez plus qu'à afficher la fonction sur l'écran, et ce, en mode RUN. Exécutez ensuite le programme DEF» «. Le programme se charge alors de lire la fonction et de la placer en ligne 1 du programme. La valeur de la fonction sera placée dans la mémoire F.

Listing de DEF» « :

```
2:» « RESTORE 1:A=PEEK 65325*256
+PEEK 65234+3: FOR I=0 TO 70:IF
PEEK (65200+I)<>&D POKE A+I,PEEK
(65200+I): NEXT I
3:POKE A+I,&3A,&E3,&D:END
```

Détail important, les fonctions mathématiques (log, sin, etc.) devront être saisies à l'aide de leurs touches et non à l'aide du clavier alphabétique.

Une astuce sur Casio fx 7700/7800 et 8700/8800 : le copier/coller d'un listing (Loïc FIEUX)

En utilisant la «mémoire de fonctions» (F-Mem) comme mémoire-tampon, il est possible de «copier/coller» un listing. Cela est particulièrement commode pour certaines instructions communes à plusieurs programmes. Par exemple, l'instruction Range 1,95,0,1,63,0 est commune à de très nombreux programmes graphiques. Nous allons voir comment copier ces instructions dans différents programmes. Rappelons avant tout que cette astuce ne concerne que les utilisateurs des Casio fx graphiques disposant d'une mémoire de fonctions accessible par [Shift] [0].

1) Affichez la liste des fonctions mémorisées en tapant [Mode] [1] [Shift] [0] [F4] (fonction List). Choisissons une mémoire de fonction non utilisée, arbitrairement, nous choisirons f6.

2) Plaçons les instructions à reproduire dans une zone de programmation, par exemple P5. Ainsi P5 contiendra Range 1,95,0,1,63,0 (adaptez ce petit exemple à vos besoins réels !).

3) Copions P5 dans f6. Alors que P5 est en cours d'édition (nous sommes en mode programme et les instructions de P5 sont affichées), appelons le menu de la mémoire de fonction par [Shift] [0] et copions le programme P5 en cours d'édition dans f6 en tapant [F1] [6] (Sto 6). A ce stade, une copie de P5 se trouve dans f6. C'est cette copie que nous allons pouvoir dupliquer dans différentes zones de programmation.

4) Copions f6 (qui contient une reproduction des instructions de P5) dans un programme. Par [Mode] [2] passons en mode programme. Choisissez un programme, éditez-le et placez-vous à l'aide des flèches de déplacement à l'endroit où doivent se placer les instructions de P5. Il suffit alors d'afficher le menu des mémoires de fonctions ([Shift] [0]) et de rappeler le contenu de f6 par [F2] [6] (Rcl 6).

Et voilà ! Un copier/coller dans le plus style «Macintosh» sur votre Casio fx ! Notez que f6 peut être reproduit autant de fois que nécessaire dans les zones de programmation...

Une astuce pour TI-82 : Changez le nom d'un programme (Loïc FIEUX)

Il est possible de «rebaptiser» un programme déjà saisi. Nous allons décrire les différentes étapes permettant d'appeler TOTO2 un programme déjà nommé TOTO1.

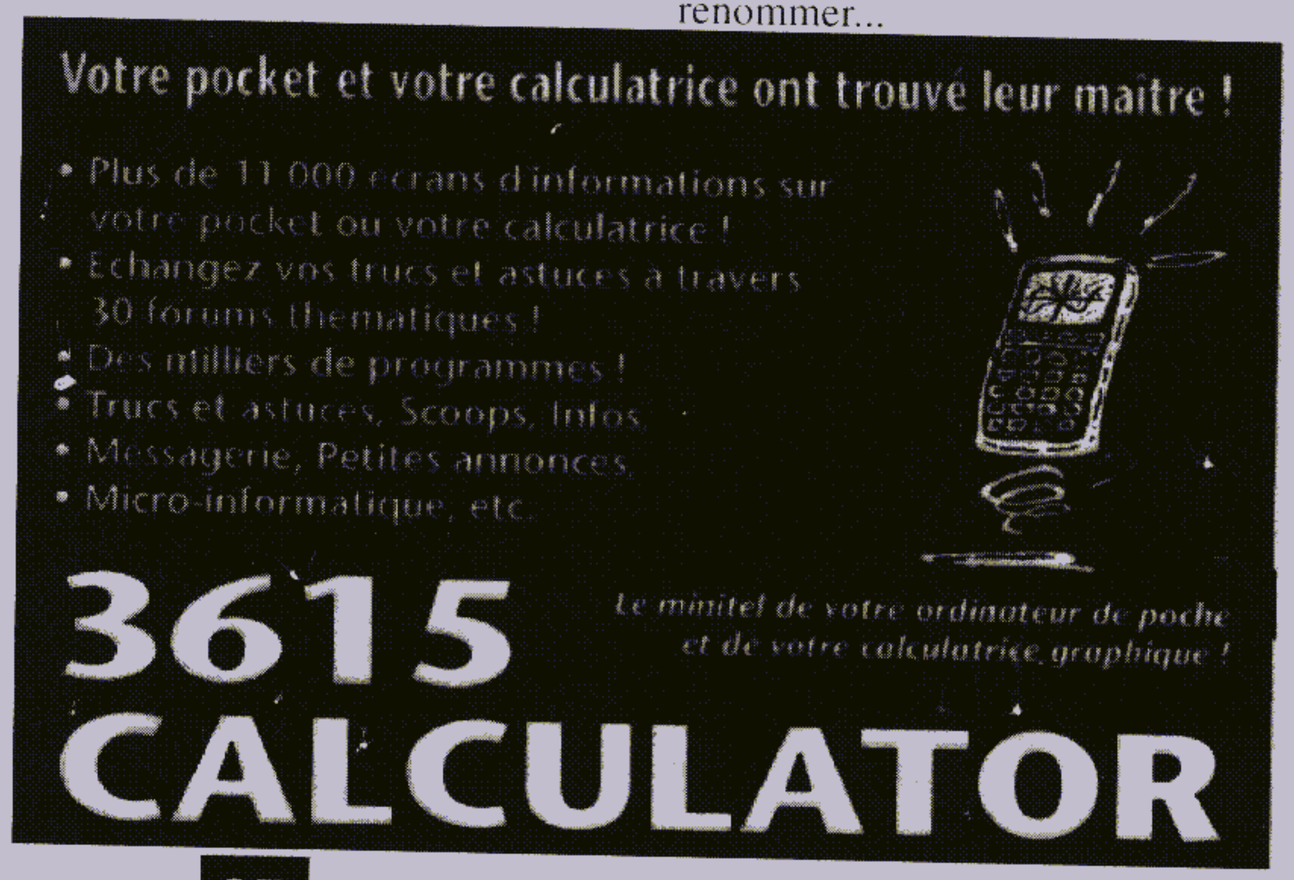
1) Appuyez sur [PRGM] pour passer en mode «programme».

2) A l'aide des flèches de déplacement, «déroulez» le menu NEW et appuyez sur [1] pour confirmer la création d'un nouveau programme.

3) Introduisez le nouveau nom, ici ce sera TOTO2. Appuyez sur [ENTER] pour confirmer.

4) Le programme TOTO2 est en cours d'édition, il ne comporte aucune instruction, cela est normal puisque c'est un nouveau programme. Nous allons maintenant copier dans TOTO2 les instructions de TOTO1, pour cela nous allons utiliser RCL. Tapons donc [2nd] [STO>] puis [PRGM], les menus «programmation» disponibles pendant l'édition d'un programme s'affichent. A l'aide des flèches de déplacement déroulons le menu EXEC et choisissons le programme TOTO1. Deux pressions sur [ENTER] entraînent la copie des instructions de TOTO1 dans TOTO2.

Il ne vous reste plus qu'à effacer TOTO1. Vous disposez maintenant d'un programme appelé TOTO2 absolument identique à TOTO1. Vous venez en fait de créer un nouveau programme comportant les mêmes instructions que le programme à renommer...



TRUCS ET ASTUCES

Sur Casio FX 6800G, le programme qui calcule DELTA prend beaucoup de place . Voici une solution plus économique. (Rodolphe MENARD).

```
"X²+BX+C=0":
"A="? --> A:"B="? --> B:
"C"? --> C disp "DELTA=":
(B²-4AC) --> D disp "R carréeDELTA"=  R
carrée D --> D disp "XI=":(-B+E) / (2A)
--> F disp "X2=":(-B-A) / (2A) --> G disp
```

Sur Casio graphique, réalisez une infinité de spirales. Modifiez éventuellement les valeurs par exemple (500,200),(5,3),(2,3) etc... (Olivier Le MÔ)

```
LBL 0
"Deg(1) ou Rad(2)" ? --> A
A=1 ==> Deg
A=2 ==> Rad
Range 0,752,900,0,496,900
MCL
1 --> I
Plot 376,248
"A="? -->A
"B="? -->B
Lbl 1
K cos (AI) -->Y
K sin (AI) -->X
Plot X + 376, Y + 248 : Line
K + 2 --> K
I + 2 --> I
I >= 250 ==> Goto 2
Goto 1
Lbl 2
"FIN" disp
Goto 0
```

modifications pour FX 6800G

```
Range 0,608,900,0,352,900
Plot 304,176
Plot X+304,Y+176
```

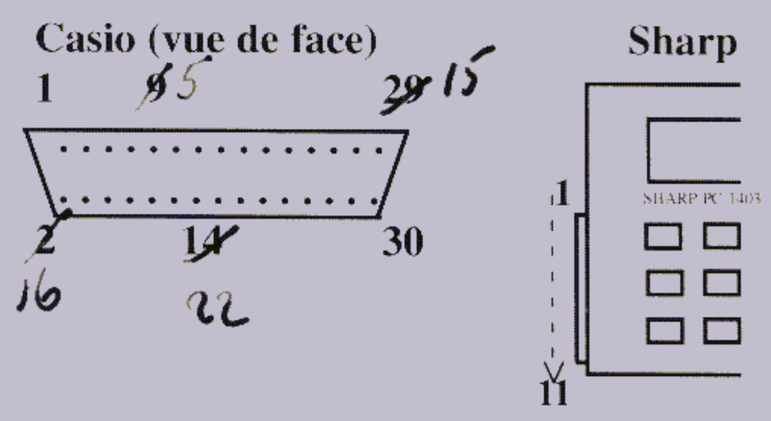
PROGRAMME SHADE 48.ASC (par Loîc FIEUX)

Ombrage d'une portion du plan sur HP48.

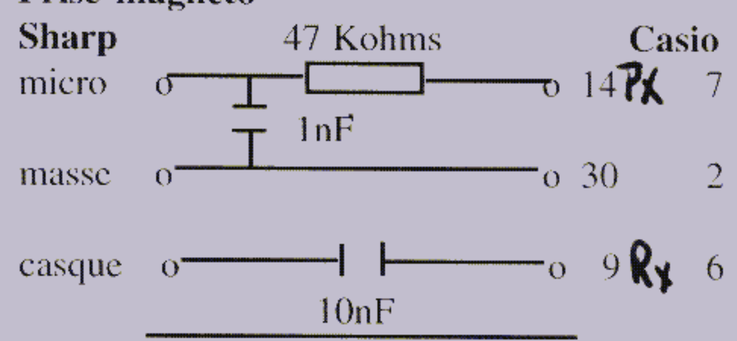
La fonction SHADE (sous-menu FCN dans l'environnement PICTURE apparaissant après le tracé d'une fonction cartésienne) est présenté à la page 22-11 du manuel des HP48 G/GX.
Si une unique fonction a été tracée, cette fonction «ombre» une portion du plan délimitée par la courbe, l'axe des abscisses ainsi que deux droites verticales (passant par les points fixés par le curseur et sa marque).
Si plusieurs fonctions sont tracées, alors la variable EQ contient une liste d'expressions algébriques.
Dans ce cas SHADE ombre une portion de plan délimitée par la courbe correspondant à l'équation sélectionnée (avec l'option NXEQ du sous-menu FCN), la courbe correspondant à l'équation qui suit l'équation sélectionnée dans la liste ainsi que deux droites verticales.
Autrement dit, si plusieurs courbes sont tracées, il n'est plus possible d'ombrer une partie du plan délimité par une courbe et l'axe des abscisses.
Pour y remédier, user de la petite astuce suivante :
Depuis PLOT, tracez vos courbes (EQ contient une liste d'équations).
Une fois les courbes tracées (vous vous trouvez dans l'environnement PICTURE), utilisez la fonction NXEQ du sous-menu FCN pour sélectionner la courbe qui doit délimiter la portion de plan à ombrer. NXEQ «sélectionne» l'une des courbes, en fait, cette fonction place l'équation correspondant à la courbe sélectionnée à la tête de la liste placée dans la variable EQ.
En appuyant sur [ON] revenez à l'application PLOT et éditez la variable EQ.
L'équation correspondant à la courbe sélectionnée se trouve en tête de la liste.
PLacez-vous juste derrière cette équation et placez en seconde position dans la liste l'expression algébrique 'O'. Celle-ci correspond à x=O, et donc à l'axe des abscisses.
Tracez les courbes (option DRAW du menu).
De retour dans PICTURE, utilisez la fonction SHADE du sous-menu FCN... Surprise !
Une astuce très simple à mettre en oeuvre pous a permis d'ombrer la portion du plan désirée.
RAPPEL : Pour marquer la position du curseur (sa «marque»), appuyez sur [x].

INTERFACES ordinateur - magnétophone pour Sharp PC 1403 et Casio FX 850P/880P (Philippe GOURVES)

Numérotation des connecteurs:



Prise magnéto



PROGRAMMES POUR CASIO. (par Karim ALAFIA)

Tracé de graphiques polaires sur graphiques (7000 - 7500 - 8000 - 8500 - ...)

```
«      GRAPH POLAIRE» : «0 MIN:»?-
>O:»0 MAX:»?->N:»EFFACE ?:»:
? -> C:C=0=>GOTO 1:CLS:LBL 1:REC( in-
sérez ici la fonction avec O pour 0,O):PLOT
I,J:O+1>O:O<>N=>GOTO 1
```

Conversion d'angles : degré en radian sous forme de fractions.

```
«CONVERSION ANGLE»:»DEG:»?->L
:»=RAD»:RAD:(L°/PI)->X:X
<>INT(X)=>GOTO1:X<>1=>X
«PI»:LBL 1:ISZ C:XC<>INT(XC) =>
GOTO 2:XC<>&XC=
«PI»:»  —»:C
```

NOTE :»<>» correspond à "différent de"

Les répertoires cachés sur HP48 (par Loîc FIEUX)

Il existe à la racine (HOME) de la mémoire utilisateur un répertoire caché (accessible par #15781h SYSEVAL). Son nom est... le nom vide ! C'est à dire ' ' ("rien" entre deux apostrophes délimitant le nom). Vous savez que l'homonimie des variables est proscrit, vous ne puovez donc pas créer un nouveau répertoire invisible dans HOME (ce répertoire invisible contient d'ailleurs des variables indispensables à la HP48). Dans un répertoire normal créé par vos soins, vous pouvez placer un sous-répertoire caché, pour cela, tapez #15777h SYS-EVAL. Pour accèder au répertoire invisible, tapez #15777h SYSEVAL EVAL. A ce stade, les objets programmes se trouvant dans le répertoire où a été placé le (sous) répertoire invisible, sont cachés. Tapez #15777h SYSEVAL PURGE pour les ressuciter ! Astuce testée sur HP48SX.

Pour PCE500

Dans un programme, les nombres sont toujours codés sous leur forme décimale ; c'est à dire 7 octets. Si vous voulez gagner de la place, tout simplement, mettez tous les nombres en hexadécimal ; exemple :
de 0 à 9 en décimal ---> 7 octets
de &0 à &9 en hexa ---> 2 octets
Ceci fonctionne aussi pour les numéros de fichiers, car ceux-ci sont codés sur 7 octets :
OPEN "E:données.dat" FOR OUTPUT AS#&1

Augmenter le nombre de pas sur une Casio graphique 7000G c'est possible !

1) enlever les piles pendant quelques minutes.
2) les remettre et allumer la calculatrice.
3) MODE 3 puis SHIFT DEL
résultat 630 pas contre 422 au départ.
Essayez aussi MODE 2 puis placez-vous sur un numéro (en fait un trait), puis appuyez sur EXE puis PRE puis MODE 2 (vous devez obtenir entre 45000 et 65000 pas !!!)

Lister la rom du PSION series 3 (Stéphane TORRES).

```
LOCAL A$(100)
A$=DIR$("ROM::*.*")
WHILE A$ <> " "
PRINT A$ : PAUSES
A$=DIR$(" ")
ENDWH
GET
ENDP
```

PROGRAMMES POUR SHARP PC1360

par Antony NEVEU

Calculez la capacité de votre mémoire

```
5«A»WAIT0:CLEAR:USING
«######.##»: GOTO 50
10POKE& C 5 0 0 , & 8 4 , & 6 0 , 0 ,
& 8 5 , & 6 0 , 0 , & 1 2 , & 5 F , & 6 1 ,
& 5 0 , & D F , & 0 4 , & 6 B , & 8 0 ,
& 3 9 , 4
20 POKE&C510,&10,&C5,&1C,&84,
&53,&11,&1D,&85,&53,&37
50 PRINT «CAPACITE =»;
60 CALL &C500
70 A=PEEK&C51C+256*PEEK&C51D
-1
80 C=A/1.78:B=C/1000
90 PRINT C;» nF»
93 USING «###.##»
95 PRINT «ou encore «;B;» micro F»
100 END
```

Un agenda pour Sharp PC1360

```
10 AUTOGOTO «A»
100 «A»: WAIT 0: CLS
120 PRINT
«************************ BLOC-
NOTES ** AGENDA *»
130 WAIT 99: PRINT «*    SHARP PC-
1360    ************************»
150 CLS : WAIT 0
160 PRINT «        M E N U»
170 PRINT «1) Ecrire un texte»: PRINT
«2) Lire un texte»: PRINT «3) Corriger
un texte»
180 CURSOR 93: INPUT C
190 IF C=1 THEN 1000
200 IF C=2 THEN 1700
1000 CLS : PRINT «        E C R I R E»:
PRINT «1) Continuer un texte»
1010 PRINT «2) Ecrire un texte»: PRINT
«3) Revenir au menu»
1020 CURSOR 93: INPUT C
1040 IF C=2 THEN 1500
1050 IF C=3 THEN 150
1055 GOTO 1540
1060 IF C<>1 THEN 1000
1070 GOTO 1530
1500 CLS : INPUT «Titre du texte ?
«;A$(Q):B=B+1:BB(B-1)=Q
1530 Q=Q+1: IF Q>150 THEN 5000
1540 CLS : INPUT A$(Q)
1550 F$= INKEY$ : IF F$=»F» LET
Q=Q+1: GOTO 3000
1560 IF F$=»» THEN 1550
1570 GOTO 1530
1700 CLS : WAIT 0: PRINT «        L I R
E»: PRINT «1) Chercher un titre»
1701 PRINT «2) Lire tous les textes»:
PRINT «3) Les noms de textes»
1702 I=0:C=0:E=0:D=0: CURSOR 94:
INPUT C: IF C=3 THEN 2500
1703 IF C=2 LET I=1: GOTO 1750
1705 CLS : INPUT «Titre du texte a lire
?    «;TT$
1710 I=I+1: IF I=BB(B) THEN 1750
1720 IF A$(BB(I-1))=TT$ THEN 1750
1730 IF A$(BB(I-1))=»» WAIT : GOTO
2000
1740 GOTO 1710
1750 E=(BB(I-1))-1
1755 IF E=((BB(I))-1) THEN 1840
1757 E=E+1
1760 CLS : PRINT A$(E)
1765 IF A$(E)=»» THEN 1850
1770 Y$= INKEY$ : IF Y$= CHR$ 4 LET
E=E-2: GOTO 1800
1780 IF Y$= CHR$ 5 GOTO 1800
1790 GOTO 1770
1800 GOTO 1755
1840 IF C=2 LET I=I+1: GOTO 1750
1850 PRINT «Fin du texte»
1855 CURSOR 72: INPUT «Fin du pro-
gramme Y/N ?»;X$: IF X$=»Y» END
1860 GOTO 150
2000 CLS : PRINT «Ce texte est inexis-
tant»: WAIT 0: GOTO 150
2500 CLS :D=D+1: IF D>B THEN 2540
2510 PRINT A$(BB(D-1))
2520 F$= INKEY$ : IF F$=»Y» LET
TT$=A$(BB(D-1)): GOTO 1710
2523 IF F$<>»» THEN 2530
2525 GOTO 2520
2530 GOTO 2500
2540 CLS : PRINT «Fin de la liste»:
GOTO 1855
3000 BB(B)=Q
3010 GOTO 1000
5000 CLS : PRINT «MEMOIRE SUR-
CHARGEE»: BEEP 1,210,90:IF INKEY$
<>»» THEN 150
5010 GOTO 5000
6000 I=0: CLS : PRINT A$(H)
6020 I=I+1
6040 CURSOR (I-1): PRINT CHR$ 249
6060 CURSOR (I-1): PRINT MID$
(A$(H),I,1)
6070 R$= INKEY$ : IF ASC R$>38 AND
ASC R$<91 OR R$=» « GOTO 10000
6080 IF R$= CHR$ 14 GOTO 6020
6100 IF R$= CHR$ 15 LET I=I-1: GOTO
6040
6120 IF R$= CHR$ 4 LET H=H-1: GOTO
6000
6140 IF R$= CHR$ 5 LET H=H+1:
GOTO 6000
6150 IF R$= CHR$ 16 THEN 8000
6170 GOTO 6040
10000 K=0: IF I=1 LET K= LEN A$(H)-
1:A$(H)=R$+ RIGHT$(A$(H)
,K):GOTO 10200
10020 K=( LEN A$(H))-I
10030 A$(150)= RIGHT$ (A$(H),K)
10040 A$(H)= LEFT$ (A$(H),(I-1))+R$
+A$(150)
10200 CURSOR 0: PRINT A$(H)
10220 GOTO 6020
60000 CLEAR : DIM A$(150)*80: DIM
BB(10)
```

TRANSFERT DE DONNEES ENTRE UN SHARP PC E500 ET UN COMPATIBLE PC

Connecter le PC E500 au PC par le câble spécial prévu à cet effet (CE130T).
Ouvrir le port série du Sharp par :
OPEN"COM:9600,N,8,1,A,L,&H1A,N,S" validez, puis CLOSE validez
Ouvrir le port série du PC compatible par:
MODE COM1:9600,N,8,1,P
Pour transférer les données :
Du Sharp vers le PC ; tapez sur le PC : COPY COM1: Nom du fichier.ext/A: validez
puis tapez sur le Sharp : SAVE "COM:" validez
Du PC vers le Sharp ; tapez sur le Sharp : LOAD "COM:" validez
puis tapez sur le PC : COPY Nom du fichier.ext/B COM1: validez
NOTA: Si vous faites souvent des transferts, il est souhaitable de sauvegarder en fichier RAM disque les commandes pour ouvrir le port série du PC E500, et en fichier Batch pour le PC compatible.
Vous pouvez également fabriquer vous-même votre câble de transfert. La difficulté viendra du connecteur Sharp (vous pouvez par exemple le fabriquer avec des pins de support à wrapper, le tout noyé dans la résine ;
voici le brochage.

Sharp	25 pins	PC 25pins	9pins	
FG	1	1		FG
TD	2	3	2	RD
RD	3	2	3	TD
RTS	4	8	1	CD
CTS	5	20	4	DTR
SG	7	7	5	SG
CD	8	4	7	RTS
RR	11	5	8	CTS
DTR	20	6	6	DSR

Quelques programmes pour SHARP PC E500 (par KLINGER Werner)

```
20 S=-1:CLEAR PX,PY:CLS :INPUT «Adresse: «;ADR ' Demander l'adresse à
partir de laquelle la mémoire sera affichée.
30 *STRT:GOTO *AFF 'début de la boucle principale
40 *CONT:CLEAR K,K$:GOSUB *TKEY ' Retour ici lorsque l'affichage ne
doit pas être modifié
50 GOSUB *CON:ADR0=ADR:ADR=ADR+((K$=»-»)-(K$=»+»))*30+((K$=»/
»)-(K$=»*»))*10 'Stockage de adr ds adr0. Si k$=»+»(«-») on incremente(dec.)
adr de 30 (3*10). Si k$=»*»(«/») on inc.(dec.) de 10 (=1 ligne
60 IF ADR>(&FFFFF-29) THEN ADR=&FFFFF-29 :CLEAR K,K$ ELSE IF
ADR<0 THEN ADR=0:CLEAR K,K$ ' Verifier que l'on veut lire une adresse
existante (0<adr<&fffff-30)
70 GOSUB *COFF:IF ADR0<>ADR THEN *STRT ' L'adresse affichee est
modifiee, il faut reafficher l'ecran
80 PX=PX+(K=15)-(K=14):PY=PY+(K=4)-(K=5) ' Modifier la pos. du curseur.
90 IF PX>9 THEN PY=PY+1:CLEAR PX:GOTO 90 ELSE IF PX<0 THEN
PY=PY-1:PX=9:GOTO 90 ELSE IF PY<0 THEN CLEAR PY :K$=»/»:GOTO 50
ELSE IF PY>2 THEN PY=2:K$=»*»:GOTO 50 ' TRAITER LES DEPASSE-
MENTS
100 *GKEY 'gestion des entrees clavier
110 GOSUB *TKEY
120 CHOIX=-(K$=» «)-(K=10)*2-(K=13)*3-(K=7)*4-(K$=»Q»)*5-(K$=»G»)
*6-(K$=»S»)*7
130 ON CHOIX GOTO *AFF,*OFF,*IN,*PIN,*QUIT,*GOTO,*SUIV
140 GOTO *CONT
200 *IN:CLEAR P$:LOCATE 0:PRINT «POKE &»;RIGHT$ («0000»+HEX$
A,5);»,»;:INPUT « ?»,P$:IF P$>»» THEN PRINT «Remplacer par «;P$;» ? «:IF
INKEY$ &1<>»O» THEN *STRT
210 IF LEN P$>1 THEN POKE A,VAL P$ ELSE POKE A,ASC P$
220 GOTO *STRT
250 *PIN:LOCATE 0:PRINT «Mode PRO de POKEs...»
:ADP=A:ADR0=ADR:ADR=ADP
260 P$=»»:PRINT «POKE &»;RIGHT$ («0000»+HEX$ ADP,5);»,&»;:INPUT
« ? «,P$:IF P$>»» THEN POKE ADP,VAL («&»+P$):ADP=ADP+1:IF
ADP<&C0000 THEN 260
270 IF ADP=ADR THEN ADR=ADR0:GOTO *STRT ELSE GOTO *STRT
300 *QUIT:END
350 *GOTO:CLEAR K,K$:G$=»»:LOCATE 0:INPUT «ALLER A L'ADRESSE
&»;G$:IF G$=»» THEN *AFF ELSE ADR=VAL («&»+G$):GOTO 60
400 *SUIV:LOCATE 0:PRINT «Quelle adresse faut-il suivre &»;:INPUT «?
«,S$:I=VAL («&»+S$):IF I>=0 AND I<=&FFFFF THEN S=I
410 GOTO *STRT
500 *AFF:IF K$=»*» THEN LOCATE 39,3:PRINT « « :FOR J=2TO 2 ELSE
CLS :FOR J=0 TO 2 ' Si «*», on n'affiche que la derniere ligne
510 LOCATE ,J:FOR I=0TO 9 'afficher la ligne de 10 octets
520 P$=CHR$ PEEK (ADR+I+J*10):LOCATE I*3:PRINT RIGHT$ («0»+HEX$
ASC P$+» «,3);:LOCATE I+30:IF P$<« « THEN PRINT «.»; ELSE PRINT P$;
530 NEXT :NEXT
540 A=ADR+PX+PY*10:P=PEEK A:PRINT «Adr: &»;RIGHT$ («0000»+HEX$
ADR,5);:LOCATE 12:PRINT «Pos: &»;RIGHT$ («0000»+HEX$ A,5)
550 GOSUB *TKEY:GOTO *CONT
600 *CON:GCURSOR (PX*18+12,PY*8+6):GPRINT «10387CFE»:GCURSOR
(180+PX*6,PY*8+6):GPRINT «FEFEFEFEFE»:A=ADR+PX+PY*10:P=PEEK
A:RETURN ' Afficher les 2 curseurs
610 *COFF:LOCATE 18,3:PRINT RIGHT$ («0000»+HEX$ A,5):LOCATE
PX*3+2,PY:PRINT « «;:LOCATE PX+30:IF P>31 THEN PRINT CHR$ P; ELSE
PRINT «.»;:RETURN ' Restorer l'ecran
650 *TKEY:L$=INKEY$ :IF L$>»» AND K$=»» THEN K$=L$:K=ASC K$ '
Stocker la 1ere touche pressee»
660 IF S>-1 THEN LOCATE 26,3:PRINT «[«;RIGHT$ («0000»+S$,5);»]=
&»;RIGHT$ («0»+HEX$ PEEK S,2);
670 RETURN
999 *:PRINT ASC INKEY$ :GOTO *
40000 *OFF:POKE &BFE00,8,0,&41:CALL &FFFDC:GOTO *CONT
```

Tables de caractères (par Werner KLINGER)

```
20 C$=»»:FOR I=160 TO (160+39):C$=C$+CHR$ I:NEXT
40 QL=PEEK &BFC93:QM=PEEK &BFC94:QH=PEEK &BFC95
50 CLS
60 POKE &BFC93,QL,QM,QH:LOCATE 0,0:PRINT C$:LOCATE 0,3:PRINT
«&»;RIGHT$ («00»+HEX$ QH,2);RIGHT$ («00»+HEX$ QM,2);RIGHT$
(«00»+HEX$ QL,2)
70 T=(INKEY$ =»-»)-(INKEY$ =»+»)+(INKEY$ =»3")*6-(INKEY$
=».»)*6:U=(INKEY$ =»/»)-(INKEY$ =»*»)+(INKEY$ =»9")*16-(INKEY$
=»6")*16
80 REM +&- DEPLACE D'UN PIXEL(=1 OCTET), /&* DE &H10 OCTETS,
3&. D'UN CARACTERE (=6 PIXELS), 9&6 DE &H100 OCTETS.
90 IF T<>0 THEN 100 ELSE IF U<>0 THEN 120 ELSE 70
100 QL=QL+T:IF QL>255 OR QL<0 THEN QL=255-(QL-T):U=(T<0)-
(T>0):GOTO 120 ELSE GOTO 60
120 QM=QM+U:IF QM<256 AND QM>=0 THEN GOTO 60 ELSE V=(U<0)-
(U>0):QM=255-(QM-U):QH=QH+V:IF QH>&2 AND QH<&F THEN 60 ELSE
QH=QH-V:GOTO 60
```

Bilan mémoire (par Werner KLINGER)

```
10 AUTOGOTO 20 ' Bilan mem, par willy
20 CLS :ON ERROR GOTO *ER1:QT$=MEM$ +»|»+»S1:»+STR$ DSKF
«S1:»:QT$=QT$+», S2:»+STR$ DSKF «S2:»
30 QT$=QT$+» Octets»:FOR I=LEN QT$ TO &23:QT$=QT$+» «:NEXT
:QT$=QT$+»DSK:»:CLS :POKE &BFCA1,&40:PRINT QT$:POKE
&BFCA1,&0
40 PRINT « E:»;DSKF &3;»Octets»
50 LOCATE &14,&1:PRINT «F:»;DSKF &4;»Octets»
60 BTEXT$ =T$:END
80 *ER1:ON ERROR GOTO *ER1:IF ERN =&4B THEN PRINT «Absent «;:IF
ERL =&14 THEN QT$=QT$+» NON.»:RESUME 30 ELSE IF ERL =&28 THEN
RESUME 50 ELSE RESUME 60
90 PRINT «FATAL ERROR #»;MID$ (STR$ ERN ,&2,&2);»AT»;ERL :STOP
```

Renseignements :
M. KLINGER WERMER
Le petit Tizin
38210 TULLINS

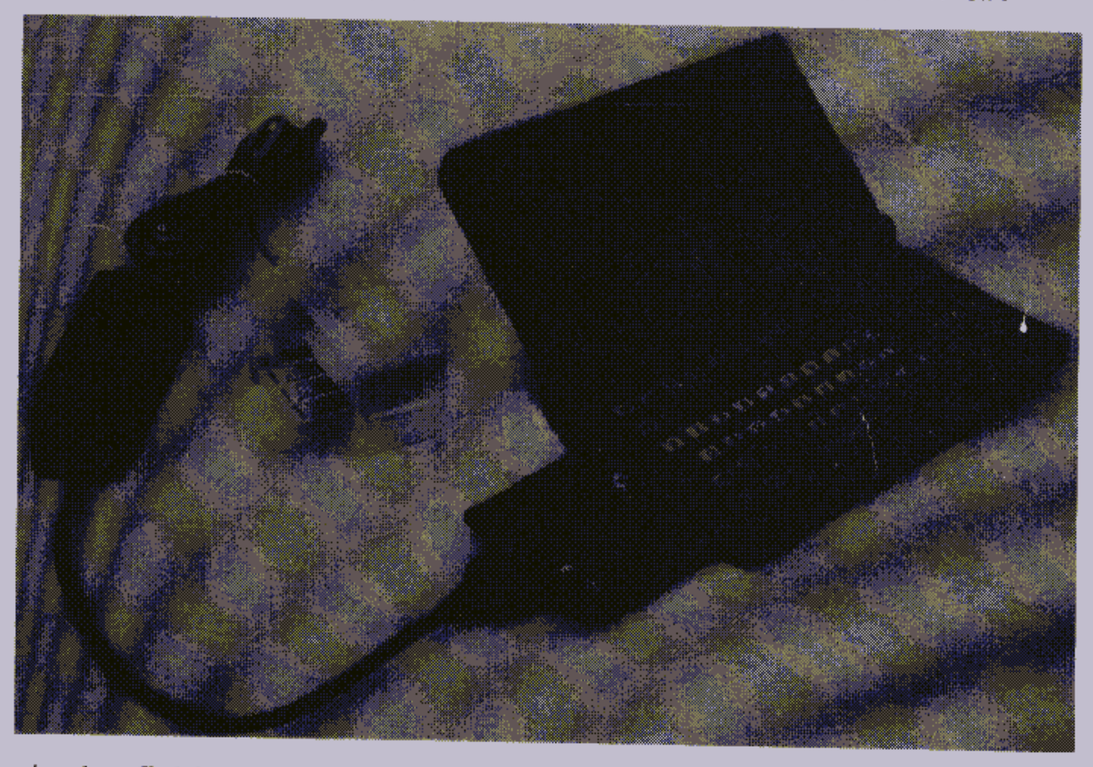
RUBRIQUE DU HP95-100LX

Nous abordons là une nouvelle rubrique uniquement destinée aux heureux (ou futurs!) posseseurs de ces petites machines géniales que sont les HP 95LX ou HP 100LX.

COMMUNIQUER AVEC UN HP 95-100LX :

Nous avons testé pour vous deux cartes PCMCIA adaptables sur l'un ou l'autre appareil. En effet, l'achat d'un portatif conduit très souvent, à terme, à se poser la question de pouvoir communiquer en "autonome". Le modem et le fax font de plus en plus partie de notre environnement immédiat. Tout repose sur le fait que la majorité des offres de matériel n'est pas prévue pour ce type d'appareils. Soit ils fonctionnent avec une carte interne, soit ils ne sont pasétudiés pour accepter l'écran ou les particuliarités des pockets. Les cartes FAX-MODEMS PCMCIA sont particulièrement bien adaptées à ces ordinateurs car elles n'occupent pas de place (rentrant complètement dans la machine) et ne pèsent pas lourd.

Le HP100LX équipé de la carte PCMCIA ThinFax



Après quelques déboires, (il faut savoir que les promesses ne correspondent pas assez souvent à la réalité ; donc méfiance ! par exemple, la carte fabriquée par Chaplet electronics "EGO FAX" ne fonctionne que dans un sens, en émission et non en réception (malgré la publicité faite pour ce produit) car dans ce domaine récent des communications, l'offre n'est pas forcément significative de qualité ; les pockets étant les "parents pauvres" pour ces fabricants, nous avons sélectionné la carte **ThinFax** fabriquée aux USA.

Le gros avantage de cette carte est que le logiciel est contenu dans une flash eprom à l'intérieur même de la carte et par conséquent n'occupe pas de place au sein du HP100. Cette flash eprom permet d'ailleurs une mise à jour éventuelle du produit au fur et à mesure qu'il s'améliore puisqu'elle est réinscriptible.

L'émission et la réception fonctionnent parfaitement. Le fax fonctionne à 9600 bauds en émission et à 4800 bauds en réception. Le modem émet ou reçoit à 2400 bauds. Ceci est très satisfaisant et correspond aux normes du groupe III, soit la plupart des fax de bureau. Dès le démarrage, on s'aperçoit que le logiciel a bien été conçu pour le HP100 ; il est très convivial et facile à mettre en oeuvre. Tout d'abord, il faut copier les fichiers autoexec.bat et config.sys de D: vers C: Ensuite, ouvrir l'éditeur de texte et supprimer le REM devant la ligne D:\bin\cic100/gen 1 de l'autoexec.bat qui permet au HP 100LX de reconnaître la carte Fax. Insérer la carte dans le port PCMCIA et faire un reset par CTRL ALT DEL pour initialiser le tout. Presser APP manager (dernière touche bleue) pour installer le logiciel minifax. Ensuite, vous n'avez plus qu'à vous laisser guider par les menus très clairs.

Sur le plan fonctionnel, on peut très bien émettre des fax (en ascii) à plusieurs correspondants choisis dans l'annuaire du 100LX. Quant à la réception, il est assez surprenant de voir apparaître par exemple de l'écriture manuscrite sur l'écran du HP ! des dessins ou des plans. Ceci est normal car on reçoit une page graphique. On peut le faire fonctionner avec un coupleur acoustique (de bonne qualité). Nous rappelons que le coupleur acoustique permet de s'affranchir d'une prise de téléphone (utile quand on est dans une cabine) ; il est constitué le plus souvent de deux "oreilles" que l'on plaquent sur l'émetteur et le récepteur du combiné téléphonique au moyen d'élastiques larges réglables. Les meilleurs coupleurs forment un bloc compact emprisonnant complètement le combiné. Le problème du coupleur est qu'il supporte mal les bruits ambiants aux alentours. Nous avons cependant réussi parfaitement une communication Fax vers TOKYO et vers MONTREAL avec un coupleur. Rappelons que la connection de la carte sur la prise téléphone nécessite une prise gigogne spéciale (fournie avec le système)

Quelques défauts remarqués : Le ThinFax ne reconnaît pas le protocole V23 (minitel) et ne fonctionne pas avec le programme buddy100. D'autre part, les minuscules et les accents français sont mal visualisés (nécessité d'écrire en majuscules pour règler le problème). Pour voir le fax reçu, le déplacement de l'écran manque un peu de rapidité.

En conclusion, cette carte, malgré quelques petits défauts est plus que satisfaisante et ce n'est pas un hasard si elle est recommandée par la société Hewlett Packard elle-même aux USA pour le HP100LX. On se met à rêver lorsque le téléphone cellulaire par satellite comme aux USA ou au CANADA fera son apparition en FRANCE ; envoyer ou recevoir un Fax au beau milieu des bois, en montagne ou sur la plage...

Le HP95LX avec la carte FAX MODEM de NEW MEDIA

Cette carte est apparue voici maintenant un peu plus d'une année. Elle possède également la particuliarité de posséder son logiciel en interne mais, se chargeant automatiquement à l'initialisation, il occupe environ 210K dans la ram du HP95. Les menus bien détaillés (en anglais !) permettent une exploitation rapide du système. Le modem émet et reçoit à 2400 bauds. Le Fax émet à 4800 bauds mais ne reçoit pas ; il est unidirectionnel. On peut facilement gérer toutes les données d'émission du modem et il est possible de créer des bibliothèques de numéros de téléphone. Cette carte n'accepte pas non plus le protocole V23 (minitel)

RUBRIQUE DU HP95-100LX

Pour relier un HP 100LX à un PC compatible, rien de plus facile !

Dans le MSDOS (5.0 et 6.0), vous trouverez les logiciels INTERLNK.EXE et INTERSVR.EXE

1) modifiez le config.sys de votre PC en ajoutant DEVICE = C:\DOS\INTERLNK.exe (si interlnk.exe se trouve dans le répertoire DOS que vous aurez créé).
2) faites l'acquisition d'un simple câble de liaison série entre PC et HP.
3) Transférez le logiciel INTERLNK dans votre HP (par exemple par le soft série intégré au HP ou par une carte mémoire PCMCIA)
4) Modifiez le config.sys de votre HP en insérant cette ligne : DEVICE = C:_dat\interlnk.exe\drives:5
5) lancez INTERSVR.exe sur votre PC
Le tour est joué !

Utiliser STACKER sur votre HP95/100:
Stacker est un merveilleux outils de compression des données, cependant, quelques précisions sont à considérer.

Vous pouvez notamment changer le ratio entre vitesse et la compression. En effet, en insérant dans le config.sys l'ordre /P= et une valeur comprise de 1 à 9, vous modifiez ces deux paramètres. Si P=1, vous augmentez la vitesse, mais vous diminuez le taux de compression. A l'inverse, si P=9, vous diminuez la vitesse mais vous augmentez le taux de compression. exemple : device=c:\stacker\stacker.com/P=1 c:\stacvol.dsk
Stacker prendra entre 39K et même 17K de mémoire pour la dernière version 4.0 (dont la compression dépasse largement les 1 pour 2 ; taux de 2,3 en moyenne)

---

6 mégas de softs pour votre HP 95LX - HP 100LX !!!

En relation avec COMPUSERVE, MICROCARDS vous propose une sélection des meilleurs freeware et shareware en direct des US, dans des domaines aussi variés que la communication, la programmation, les jeux, les utilitaires, les applications Lotus, Mail, le traitement de textes, la musique etc...
La documentation est fournie en anglais.
Ces programmes sont garantis strictement testés sans virus connus.

1) le pack N°1 de 3 disquettes 3,5 pouces 1,44MB 170F ttc (port retour et disquettes comprises)
2) le pack N°2 de 5 disquettes 3,5 pouces 1,44MB 250F ttc (port retour et disquettes comprises)
supplément de 30F pour des disquettes 3,5 pouces 720K ou autre format (le préciser dans votre commande)

Et pour ceux qui, malheureusement ne possèdent pas de PC de bureau, MICROCARDS peut vous fournir ces softs sur carte mémoire.
1) vous expédiez votre (vos) carte(s) mémoire PCMCIA vierge (type I ou II) SRAM (après avoir vérifié l'état de la pile) par PTT en colissimo recommandé R3 ; on vous la(les) chargera pleine de softs et on vous la(les) renverra en quelques jours par PTT colissimo recommandé pour 150F ttc (+100F par carte supplémentaire) port retour compris. (précisez si vous possédez un HP95, un HP100 ou un Omnibook et si vous désirez plutôt des utilitaires, des jeux, des applications, ou un mélange des trois)
2) en cas d'achat d'une carte mémoire PCMCIA pour HP95-100 ou Omnibook chez MICROCARDS, celle-ci sera fournie remplie de softs sans augmentation de prix.
UTILISEZ LE BON DE COMMANDE EN DERNIERES PAGES DE POCHE COMPUTERS. Merci

---

Nouvelles cartes PCMCIA langages ou softs divers. (pour HP95-100 ou Omnibook)

Nombreux traducteurs de langues différentes (Français - Anglais - Allemand - Espagnol - Latin - Italien - Russe - etc...)

GraphMaster (créez vos graphes et tableaux divers)

MusicBases (base de données sophistiquée pour amateurs de musique)
HoroScopicsII (calcul d'horoscopes professionnel

Travelware (organisez, planifiez votre voyage)

CoinBase (super base de données pour collectionneurs)

Bearings (repérage pour pilotes et radio amateurs)

Gravitor (suiver les étoiles et les planètes)

Tarotmaster (la fortune est à vous !)

Runemaster (la divination à votre portée !)

FinancialSuper (Calculs financiers en tous genres)

AWELE SUR PSION SERIES 3/3A

par le Docteur Maurice TALAZAC

Jeu de réflexion par excellence, l'AWELE est à l'Afrique ce que les Echecs sont à l'Europe, et le Go à l'Asie. Connu sous d'innombrables variantes (WARI, MANCALA, SOLO...), son aspect mathématique en fait un jeu idéal pour la programmation.

Les règles:
Le jeu se pratique à deux, sur une table creusée de deux rangées de 6 alvéoles. Chaque alvéole contient 4 pions (on utilise de très beaux coquillages, des graines ou des cailloux). Chaque joueur possède une rangée. Pour jouer, il prend tous les pions d'une alvéole quelconque de sa rangée, et les distribue un par un successivement dans les alvéoles voisines, dans le sens inverse des aiguilles d'une montre.

```
    nord
4 4 4 4 4 4     Le joueur sud joue sa    4 4 4 4 5 5
4 4 4 4 4 4     quatrième case.......>   4 4 4 0 5 5
    sud    (cases numérotées de gauche à droite)

4 4 4 4 5 5     Le joueur nord joue sa   0 4 4 4 5 5
4 4 4 0 5 5    sixième case............> 5 5 5 1 6 6
```

Si le dernier pion tombe dans une alvéole du côté adverse, et que celle ci contient déjà seulement un ou deux pions, (ce qui fait deux ou trois pions une fois celui du joueur posé), il prend tous les pions contenus dans l'alvéole, et les sort du jeu. Si l'alvéole immédiatement voisine à droite contient aussi deux ou trois pions, il les prend aussi, et ainsi de suite jusqu'à tomber sur une aléole contenant moins de deux ou plus de trois pions.

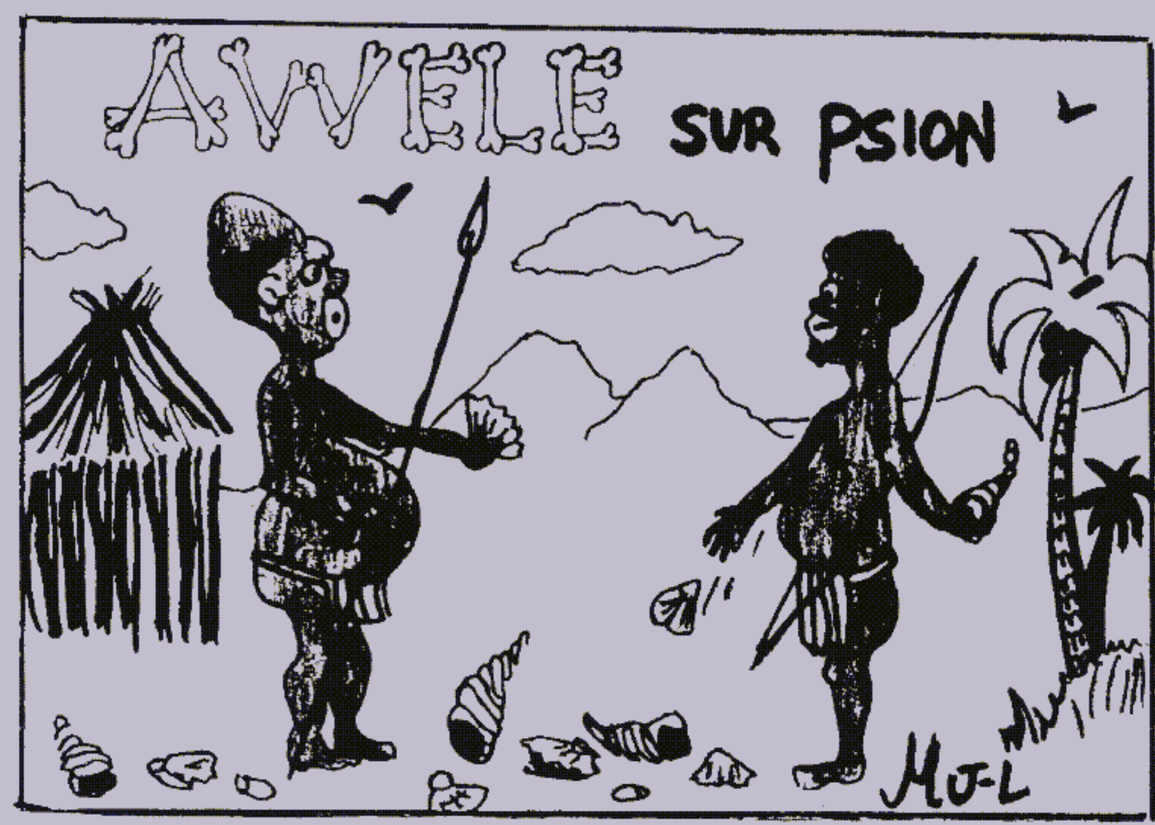
```
4 0 6 3 1 1     Le Joueur nord joue sa   5 1 0 3 1 1
2 4 3 1 3 3     quatrième case..........> 3 5 4 0 3 3
et vide la quatrième case du joueur sud.

4 2 1 1 0 4     Le joueur sud joue sa    4 0 0 0 1 5
2 3 0 0 6 1     cinquième case.......>   2 3 0 0 0 2
et vide les 5è, 4è, et 3ème case du joueur nord.
```

Ces règles simples sont complétées par deux précisions :
- Si les cases de l'adversaire sont toutes vides, on est obligé de jouer un coup qui en remplisse une, si c'est possible. Sinon, le jeu est terminé.

-Si on joue une case contenant plus de onze pions, on la saute quand on revient sur elle en distribuant les pions, de façon à ce que toute case jouée soit toujours vide après son tour.
Le jeu est terminé en cas de blocage, ou quand la différence de score entre les joueurs est supérieure au nombre total de pions restants sur le plateau. Le score est établi en additinnant pions pris, et pions restants dans chaque camps.

Le programme:
Ecrit pour un Séries 3A, il peut être effectué sur un Séries 3 en modifiant les valeurs des commandes graphiques comme indiqué dans les lignes REM.
Le plateau de jeu est représenté par une variable tableau A%(X%). X% représente le numéro de case, de 1 à 6 pour le joueur, et de 7 à 12 pour l'ordinateur. A%(X%) représente le contenu de la case numéro X%.
Le premier coup est au choix un coup du joueur, ou un coup au hasard de l'ordinateur.
La réflexion de l'ordinateur est une recherche exhaustive à deux niveaux. Aprés mise en mémoire des positions dans le tableau R1%(X%), l'ordinnateur étudie chacun de ses coups possibles (procédures VALID, puis MVT) en attribuant à chaque position un score SC1% (procédure EVAL). Il fait de même ensuite pour chacune des réponses possibles du joueur, avec le tableau R2(X%), et le score SC2%. Le coup XMAX% permettant le meilleur score SCMAX%=SC1%-SC2% est ensuite joué (procédure MVT), les prises effectuées (procédure PRISE) et le résultat affiché. Aprés vérification de l'éventuelle fin du jeu (procédure TERMIN), le cycle recommence.
La procédure EVAL attribue un score en fontion du nombre de prises possibles (procédure PRISE), et du nombre de cases prenables, contenant 1 ou 2 pions.

Le programme pourra être perfectionné en affinant la procédure d'évaluation, et en augmentant le niveau de réflexion, avec les méthodes classiques d'élagage sélectif d'arborescence (algorithme alpha/bêta). En attendant, l'ordinateur est déjà assez difficile à battre, pour vous permettre un entraînement sérieux avant votre prochain voyage en Afrique.




```
PROCAWELE:

REM AUTEUR Maurice TALAZAC ************
REM PROGRAMME ECRIT POUR PSION SERIES 3A
REM POUR SERIES 3, UTILISER LES COORDONNEES
REM D'AFFICHAGE MENTIONNEES EN REM

GLOBALA%(12),R1%(12),R2%(12),X%,XMAX%,XM%,Y%,VLD%,
LIM%,Z%,G%,GLOBALSCMAX%,SCORE%,SC1%,SC2%,
GAINO%,GAINJ%,
GAIN%,FINI%
GLOBAL VLD1%,VLD2%
DEBUT::
 REM INITIALISATION ********************
X%=0 :GAINJ%=0 :GAINO%=0 :RANDOMIZE SECOND
DO
X%=X%+1:A%(X%)=4
UNTIL X%=12
 CLS :REM AFFIGAGE TABLEAU***************
 GSTYLE 0 :GTMODE 3
 GAT 40,15: REM 20,9 POUR SERIES 3
 GFILL 281,97,0 :REM 140,48,0
 X%=0
 DO
 GAT 63+X%46,13 :REM 31+X%*23,8
 GPRINT 12-X%
 GAT 63+X%*46,126 :REM 31+X%*23,65
 GPRINT 1+X%
 GAT 45+X%*46 20 :REM 22+X%*23,11
 GFILL 41,41,1 :REM 21,21,1
 GAT 45+X%*46,66 :REM 22+X%*23,34
 GFILL 41,41,1 :REM 21,21,1
 X%=X%+1
 UNTIL X%=6
 GSTYLE 9 :REM GSTYLE 0
 AFF:
 GAT 5,155 :REM 2,79
 GPRINT «VOULEZ-VOUS COMMENCER      (O/N)»
 G%=GET
 IF G%=78 : X%=12-INT(RND*6) :XMAX%=X% :GOTO BEGIN :ENDIF

 JOUEUR:: :REM COUP DU JOUEUR ***********
 LIM%=6
 TERMIN:
 IF FINI%=1 : GOTO FINJEU :ENDIF
 GAT 240,155 :REM 2,79
```

AWELE SUR PSION SERIES 3/3A

```
GPRINT «.  Tapez votre numéro de case   :»,
G%=GET
IF G%<49 OR G%>54 :GOTO JOUEUR  : ENDIF
X%=G%-48
VALID: :REM VERIFIE VALIDITE DU COUP****
IF VLD%=1
GOTO JOUEUR
ENDIF
GAIN%=0
MVT: :REM EFFECTUE LE COUP *************
PRISE:
GAINJ%=GAINJ%+GAIN%
AFF:
GAT 5,155 :REM 2,79
GPRINT «Vous avez joué  «,X%,» EN «,XM%,».  Tapez ESPACE pour coup
 ordinateur .»
RETOUR::
G%=GET :IF G%<>32 :GOTO RETOUR :ENDIF
GAT 195,155 :REM 2,79
GPRINT «.    REFLEXION ...»
ORDIN::  :REM COUP DE L'ORDINATEUR ****
LIM%=0
TERMIN:
IF FINI%=1 : GOTO FINJEU :ENDIF
X%=0 :SCMAX%=-100
DO :REM SAUVEGARDE POSITION INITIALE***
   X%=X%+1 :R1%(X%)=A%(X%)
 UNTIL X%=12
 X%=6 :REM REFLEXION ORDINATEUR 1ER NIVEAU**
 DO :REM TESTE CHAQUE COUP*****************
   X%=X%+1
   Y%=0 :REM RESTITUE POSITION INITIALE*****
   DO
 Y%=Y%+1 :A%(Y%)=R1%(Y%)
UNTIL Y%=12
VALID: :REM SI COUP POSSIBLE **************
VLD1%=VLD%
IF VLD1%<>1
  MVT: :REM ON EFFECTUE LE COUP ***********
  EVAL: :REM ON L'EVALUE *****************
  SC1%=SCORE%
    REM REFLEXION 2E NIVEAU ***************
   Z%=0 :LIM%=6 :SC2%=0
   DO :REM SAUVEGARDE POSITION INTERMEDIAIRE**
     Z%=Z%+1 :R2%(Z%)=A%(Z%)
   UNTIL Z%=12
   Z%=X%
   X%=0
   DO :
  REM TESTE COUPS JOUEUR ************
      X%=X%=1
      Y%=0 :
 REM RESTITUTE POSITION INTERMEDIAIRE
      DO
       Y%=Y%+ :A%(Y%)=R2%(Y%)
      UNTIL Y%=12
      VALID: :REM SI COUP POSSIBLE *******
      VLD2%=VLD%
      IF VLD2%<>1
       MVT: :REM ON L'EFFECTUE **********
       EVAL: :REM ON L'EVALUE ***********
      IF SCORE%>SC2% :SC2%=SCORE% :ENDIF
     ENDIF
    UNTIL X%=6
   X%=Z% :REM RESTITUE X ******************
   LIM%=0
   SCORE%=SC1%-SC2% :REM DETERMINE SCORE*****
   IF SCORE%=SCMAX% :XMAX%=X% :ENDIF
   IF SCORE% >SCMAX% :SCMAX%=SCORE% :XMAX%=X% :ENDIF
  ENDIF
UNTIL X%=12 :REM FIN REFLEXION ***********
 X%=XMAX% :REM ORDINATEUR JOUE ************
  Y%=0 :REM RESTITUE POSITION INITIALE****
  DO
  Y%=Y%+1 :A%(Y%)=R1%(Y%)
  UNTIL Y%=12
BEGIN: :
GAIN%=0
MVT:
PRISE:
GAINO%=GAIMO%+GAIN%
AFF:
GAT 05,155 :REM 2,79
GPRINT «L'Ordinateur a joué «,XMAX%,» en « ,XM%,»          ,»
GOTO JOUEUR
FINJEU: :REM LE JEU EST TERMINE***********
X%=0 :REM COMPTE PIONS RESTANTS***********
 DO
 X%=X%+1
 IF X%<7 :GAINJ%=GAINJ%+A%(X%) :ENDIF
 IF X%>6 :GAINO%=GAINO%+A%(X%) :ENDIF
 UNTIL X%=12
 GAT 5,155 :REM 2,79
 IF GAINJ% > GAIN0%
 GPRINT «* VOUS AVEZ GAGNEZ PAR «,GAINJ%,»  A  «,GAINO%,».
Autre partie ? (O/N)
 ENDIF
 IF GAINO% > GAINJ%  GPRINT «* L'ORDINATEUR A GAGNE PAR
«,GAINO%,» A «,GAINJ%,».
Autre partie ? (O/N)
 ENDIF
 IF GAINJ% = GAINO%
 GPRINT «*          MATCH NUL...                    .AUTRE
parti ? (O/N)
 ENDIF  G%=GET
 IF G%=79 : GOTO DEBUT :ENDIF
ENDP
PROC AFF: :REM AFFICHAGE DE CHAQUE COUP ***
 LOCAL K%
 K%=0
 DO
  GAT 62+K%+46,51 :REM 31+K%*23,26
  IF A%(12-K%)<>0
   GPRINT A%(12-K%)
  ELSE GPRINT «      « :REM «
  ENDIF    GAT 62+K%*46,97 :REM 31+K%*23,48
  IF A%(1+K%)<>0
   GPRINT A%(1+K%)
  ELSE GPRINT «      « :REM «    «
  ENDIF    K%=k%+1
 UNTIL K%=6
 GAT 340,51 :REM 170,25
 GPRINT «<  ORDIN. :» ,GAINO%
 GAT 340,100 :REM 170,50
 GPRINT «<  JOUEUR:»,GAINJ%
ENDP
PROC MVT: :REM EFFECTUE MOUVEMENT********
 LOCAL LONG%, K%,Z%
 LONG%=A%(X%) :K%=0 :Z%=X%
 A%(X%)=0
 DO
  K%=K%+1
  X%=X%+1    IF X%>12 :X%=X%-12 :ENDIF
  A%(x%)=A%(X%)+1
  IF X%=Z% : A%(X%)=A%(X%)-1 :LONG%=LONG%+1 :ENDIF
 UNTIL K%=LONG%
 XM%=X% :REM XM% =DERNIER CASE ATTEINTE*****
 X%=Z%
ENDP
PROC VALID: REM VALIDITE DU COUP **********
 VLD%=0
 REM SI CASE VIDE?
 IF A%(X%)=0 :VLD%=1 :ENDIF
 REM OU SI ADVERSAIRE VIDE?
 IF A%(LIM%+1)=0 AND A%(LIM%+2)=0 AND A%(LIM%+3)=0 AND
A%(LIM%+4)=0 AND A%(LIM%+5)=0 AND A%(LIM%+6)=0
```

AWELE SUR PSION SERIES 3/3A

```
  IF X%+A%(X%) < 13-LIM% :REM ET COUP NE LE REMPLIT PAS
    VLD%=1  :REM COUP NON VALIDE**********
  ENDIF
 ENDIF
ENDP
PROC EVAL: :REM DONNE UN SCORE A POSITION ***
LOCAL K%
 PRISE: :REM EN COMPTANT LES PRISES*********
 K%=0 :REM ET LES CASES DE1 ET 2 PIONS******
 DO
  K%=K%+1
  IF K%<7 AND (A%(K%)=1 OR A%(K%)=2)
:SCORE%=SCORE%+A%(K%) : ENDIF
  IF K%>6 AND (A%(K%)=1 OR A%(K%)=2) :SCORE%=SCORE%-
A%(K%) : ENDIF
 UNTIL K%=12ENDP
PROC PRISE: :REM VERIFIE SI PIONS PRIS ****
 LOCAL G%
 SCORE%=0: GAIN%=0  G%=XM% :REM DERNIERE CASE REM-
PLIE**********
  IF LIM%=0 AND G%>6 :GOTO FIN :ENDIF
  IF LIM%=6 AND G%<7 :GOTO FIN :ENDIF
  IF A%(G%)=2 OR A%(G%)=3
   SCORE%=SCORE%+10*A%(G%)
   GAIN%=GAIN%+A%(G%)
   A%(G%)=0
    IF G%>1+LIM%      DO      G%=G%-1
      IF A%(G%)<>2 AND A%(G%)<>3          BREAK
      ENDIF        SCORE%=SCORE%+10*A%(G%)
      GAIN%+GAIN%+A%(G%)
      A%(G%)=0
     UNTIL G%=1+LIM%
    ENDIF
   ENDIF  FIN::
ENDP
PROC TERMIN: :REM VERIFIE SI JEU TERMINE ***
 LOCAL SOMME%  X%=6-LIM% :FINI%=1 :REM SI AUCUN COUP
VALIDE, FIN*
 DO    X%=X%+1    VALID:
  IF VLD% =0 :FINI% =0 :ENDIF
 UNTIL X%=12-LIM%
 REM SI DIFFERENCE SCORES > NB DE PIONS RESTANTS
 SOMME%=0 :X%=0
 DO
  X%=X%+1
  SOMME%=SOMME%+A%(X%)
 UNTIL X%=12
 IF SOMME%<GAINJ%-GAINO% OR SOMME%<GAINO%-GAINJ%
  FINI%=1
 ENDIF
ENDP
```

IMPRESSION SERIE SUR IMPRIMANTE PARALLELE pour PSION II (par Michel SANTURENNE)

Posseseurs d'un ATARI 520ST, et d'une interface série, réjouissez-vous ! Vous allez pouvoir imprimer à partir de votre PSION II. Une fois chargé en mémoire, il suffit d'appeler le programme pour imprimer n'importe quel fichier (agenda, note, cpl ...) ; et celui-ci reste actif tant que l'on ne touche pas à la souris. Ecrit en GFA basic 3.5, cela ne doit pas poser beaucoup de problèmes pour l'adapter à une autre machine.

```
~XBIOS (31,3,1,2,L:-1)      ; 9600 bauds
deb:
PRINT : PRINT : PRINT : PRINT
PRINT  "                    Attente d'impression"
DO
   a= inp?(1)
LOOP UNTIL a<>0
PRINT : PRINT : PRINT : PRINT : PRINT
PRINT "                     Début d'impression"
PRINT : PRINT
PRINT "      appuyer sur un bouton de la souris en fin d'impression"
DO
     a$=INPAUX$
      LPRINT a$;
LOOP UNTIL MOUSEK
CLS
GOTO deb
```

Version accessoire écrit en GFA Basic 3.5 et compilé en 3.5, mais peut être utilisé en tant que programme en GFA 2.0 (listing précédent)

```
$m15000   ;minimum mémoire
ap_id&=APPL_INIT()
IF ap_id&
   PRINT "impression PSION chargée"
   VSYNC
   VSYNC
   me_id&=MENU_REGISTER(ap_id&,"    PSION")
   DO
      ~EVNT_MESAG(0)
      IF MENU(1)=40
deb:
  ALERT 1,"imprimer",1,"oui : non",x%
  IF x%=2
          QUIT
  ENDIF
  ~XBIOS(31,3,1,2,L:-1)                ; 9600 bauds
  DO
          a=INP?(1)
  LOOP UNTIL a<>0
  DO
          a$=INPAUX$
          LPRINT a$;
  LOOP UNTIL MOUSEK            ; attente clic souris
  ALERT 2," AUTRE IMPRESSION ?",1,"OUI : NON",b
  IF b=2
          QUIT
  ENDIF
```

Michel SANTURENNE est pilote d'avion. Il a également réalisé un logiciel sur PSION II, de navigation aérienne, apparemment de première qualité dont voici quelques caractéristiques concernant les commandes et calculs. (les initiés comprendront ... eux !!!)
Atmosphère - Plan vertical - Altimétrie - Mécanique du vol - Navigation- Lever et coucher du soleil - QDR - Centrage avion - Vitesse etc... IFR - CONVERSIONS - Simulation complète de la machine à calculer spécialisée aéronautique PATH FINDER CX1 - Gestion des balises - Way points et caractéristiques des aérodromes - Impression possible sur imprimante sur forme de liste ou de tableau.

Espérons voir ce programme dans POCHE COMPUTERS bientôt...

essai : UN LECTEUR DE DISQUETTES POUR PSION series 3/3A

Un lecteur de disquette pour le PSION, est-ce possible ? Eh bien oui, nous en avons testé un ; celui commercialisé par MICROCARDS depuis le début de cette année. Vous me direz : à quoi peut bien servir un boittier lecteur de disquettes quand on a acheté le PSION series justement parcequ'il est tout petit ? Ceci peut paraître à première vue une contradiction fondamentamentale...

Mais à la réflexion, on s'aperçoit vite que la conservation des données par SSD coûte cher sur le PSION alors qu'une disquette de 1,44MB coûte moins de 10F pièce ! De plus, quel plaisir de travailler chez soi ou au bureau avec un système fiable et puissant ! Dans ce cas, la portabilité du PSION n'est pas son meilleur atout. Au contraire, la recherche de puissance, de versatilité et d'échanges possibles sont primordiaux.

Puis, on enlève le fil à la patte, et le PSION redevient une merveilleuse petite machine autonome.

1)Installation:

Le lecteur de disquettes fonctionne sur secteur (une alimentation spéciale est fournie) et la transmission des données passe par la sortie série du PSION series 3 ou 3a. Il est donc nécessaire de posséder au préalable le "sérial link"

Cependant, nul besoin d'autre chose ; en effet, on utilise le soft du sérial link pour communiquer avec le lecteur. Après installation du soft "Comms" dans le PSION, la connectique est très simple. D'une part la liaison sérial link (1 connecteur 9 points et 1 25 points sont prévus), et d'autre part, la liaison secteur par l'adaptateur. L'installation du soft Comms.app est décrite dans le manuel du Psion et est très simple. Ensuite, créer un icône supplémentaire.

Il faut maintenant règler les paramètres de communications qui sont les suivants : 9600 bauds - 8 bits - 1 bit de stop - sans parité - Ignore parité ; oui - Xon/xoff ; Off - Rts/Cts ; Off - Dsr/Dtr ; Off-Dcd;Off.

Puis, l'accès au menu principal apparaît après avoir tapé "S3" (comme series 3) et répondu à quelques questions pour vérifier la connection : adaptateur et lecteur.

Sur l'écran du PSION, vous avez alors le chox de :

- Lister les fichiers contenus sur disquettes
- Sauvegarder un fichier sur disquette
- Charger un fichier depuis une disquette
- Afficher le directory
- Utilitaires (Formater - changer l'heure ou la date - renommer un fichier - effacer un fichier)

Toutes ces fonctions sont accessibles directement en appuyant sur une touche. Les transferts sont moins rapides qu'à partir d'une SSD, c'est évident, mais cela reste correct. Le lecteur accepte deux types de disquettes : des 720K ou des 1,44MB. On peut les formater indifféremment (environ 1minutes 30 pour la première et 2 minutes pour la seconde).

Le format est compatible PC. Pour transférer par exemple des données de votre PSION series 3/3a vers un PC compatible, rien de plus simple.

a) sauvegarder vos données sur PSION dans un format PC compatible (par exemple au format ASCII ".text")

b) transferer vos données vers le lecteur de disquettes.

c) introduisez cette disquette dans un PC

d) importez les données à partir de votre traitement de textes ou base de données préférée. Quant au tableur, l'extension .WK1 est reconnue par la majorité des tableurs PC, sous Dos ou sous Windows.

Pour terminer, voici les spécifications techniques du lecteur :

Formatage des disquettes : MS-DOS 3.3 et plus compatible

Capacité : 1,44MB par disquette (720K supportée aussi)

Dimensions : 185mm x 117mm x 43mm

Poids : 1kg

Alimentation 6-9volts 1A

Fichiers : compatibles PC (224 maximum en racine)
(224 sous directories maximum)

Dernière chose importante, MICROCARDS a considérablement baissé les prix du lecteur. Celui-ci devenant beaucoup accessible et entrant en concurrence directe avec le système des SSD.

JL LECOMTE

Interface de commande 8 sorties pour PSION series 3a

par Patrick DOUGUET

Nous allons abordr un domaine peu exploré dans le monde de la micro informatique ; la Robotique. Depuis peu, les micro sont doués de la parole et oh! surprise, le PSION ne faillit pas à cette engouement. Dans un avenir proche, il obéira à la voix, car contrairement à certains qui tablent sur la reconnaissance de l'écriture, PSION annonce déjà un nouveau modèle à commande vocale ! Rendez-vous dans deux ans... En attentant, dans quelle mesure un ordinateur peut-il être considéré comme intelligent ? Quelles sont ses capacités et ses limites ? En un mot, que peut-on attendre de la robotique. C'est à ces diverses questions que nous allons tenter de répondre par la pratique.
Un ordinateur peut être considéré comme l'élément intelligent, le cerveau, en quelque sorte, du robot. Il y a peu, un IBM PC prenait un volume non négligeable, notre PSION en a les capacités, mais pas l'encombrement; or pour le transformer en robot que lui manque-t-il ?
Tout d'abord la capacité d'émettre des signaux à destination de son environnement. C'est donc à ce premier problème que nous allons nous consacrer. Pour tenir compte de la grande variété de systèmes informatiques, nous avons choisi d'utiliser la prise parallèle qui existe sur tous micros.
Dans la suite de l'article nous nous adressons plus particulièrement aux heureux possesseurs de PSION série 3a, mais l'adaptation logicielle à d'autres machines ne pose, en principe, aucun problèmes.

Notre réalisation va permettre de commander une première série de huit relais, qui permettrons d'envisager toutes sortes d'applications (domotique, scientifique, robotique, ludique, ...). Pour cela regardons le schéma ci-après. Tout d'abord le PSION a besoin d'une interface parallèle qui se termine par une fiche Amphénol à 36 broches classique. Notre premier soucis sera de trouver une prise identique mais femelle, les autres choisirons une fiche type DB25, les indications de numéro de broche étant valables dans les deux cas.
Dans un premier temps, nous relierons les plots 11 et 25 (masse), pour dire à l'ordinateur que le périphérique est prêt à recevoir ses ordres. Les plots 2 à 9 servent, en théorie, à transmettre les données correspondant à la transmission de caractères vers une imprimante, ce sont ces données que nous allons détourner de leur fonction pour commander les relais. chaque sortie de DATA est reliée à un photo-coupleur (2N35 ou équivalent) afin d'isoler électriquement le PSION des circuits commandés (surtout s'ils véhiculent du secteur 220 V). La sortie du photo-coupleur alimente le relais. C'est tout ce qu'il nous faut pour construire notre interface universelle 8 sorties. Un zest de logiciel en basic OPL et voilà notre robot prêt à entrer en fonction. Rappelez-vous qu'il n'y a pas de mauvais robots, mais seulement de robots mal programmés.

PROGRAM A PSION

Ce programme permet d'allumer ou d'éteindre une sortie quelconque sans affecter l'état des autres sorties. Pour visualiser la sortie on optera pour une série de diodes led.
Le logiciel utilise l'OPL, qui bizarrement, ne gère pas les fonctions **in** et **out** que l'on trouve habituellement en qbasic sur IBM. Néanmoins une astuce va nous permettre de pallier à cette déficience.
Pour imprimer un caractère, on envoie un code compris entre 0 et 255 (ex. le code chr$(65) imprime la lettre A), or ce code vient lui-même du binaire. Là réside la solution. 01010111 en binaire donne 87 en décimal, cela représentera aussi l'état des sorties : Sorties n° 8, 6, 4 inactives et sorties n° 1, 2, 3, 5, 7 actives. Il ne reste plus qu'à utiliser correctement cette conversion pour commander indépendamment chaque relais.

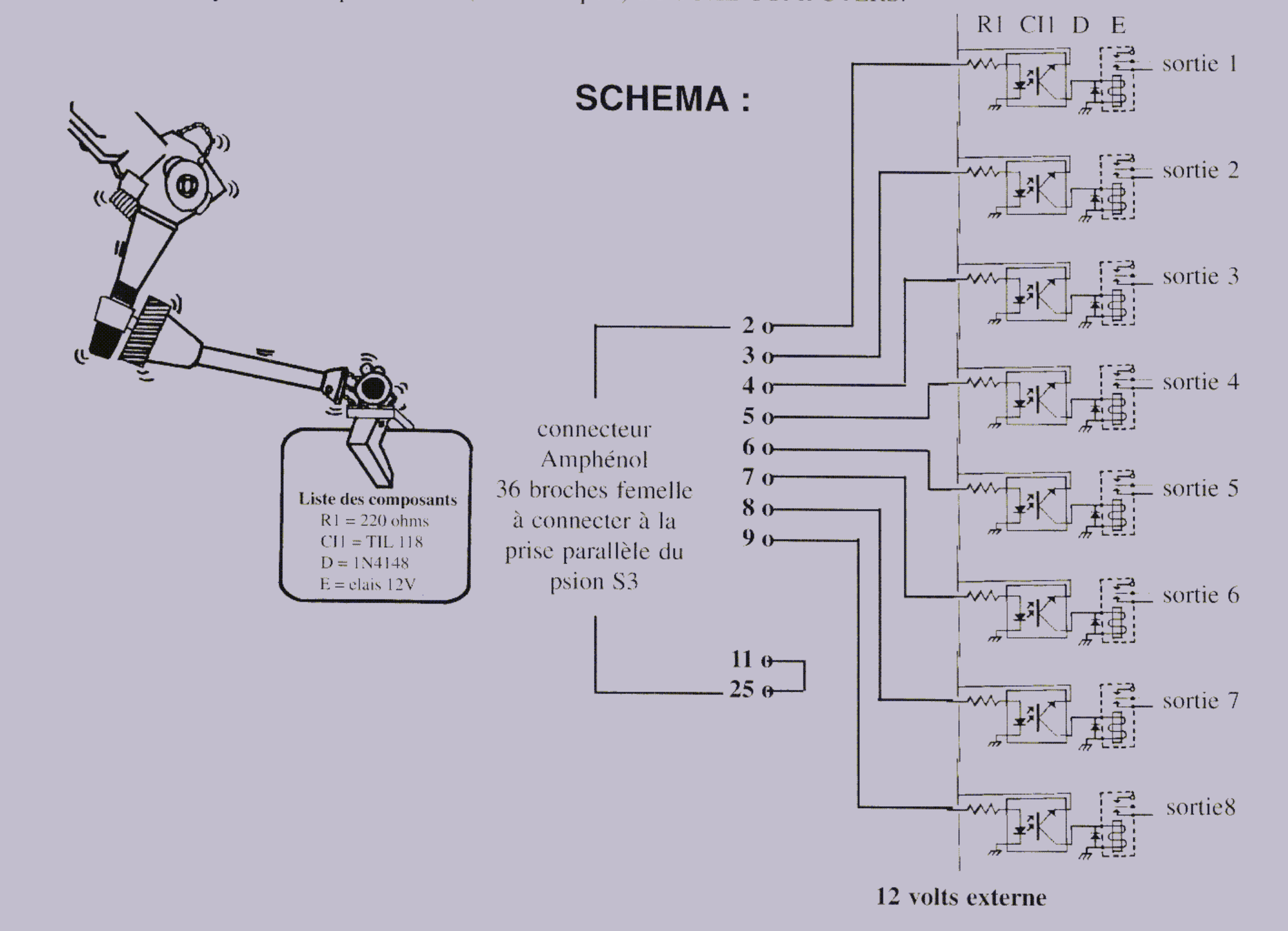
```
PROC. SORTIE:
LOCAL S(8)
LOCAL A(8)
LOCAL n,e,x
        DO
                CLS
                PRINT «CHOIX DE LA SORTIE A COMMAN
                        DER (1 à 8):»
                INPUT n
                PRINT «ETAT DE LA SORTIE : 1=ON
                        0=OFF»
                INPUT e
                S(n)=e
```

Interface de commande 8 sorties pour PSION series 3a

```
x=S(1)+2*S(2)+4*S(3)+8*S(4)+16*S(5)+32*S(6)+64*S(7)+
128*S(8)
        PRINT «JE COMMANDE LA SORTIE : «;n
        LPRINT CHR$(x)
        UNTIL x=10
LCLOSE
ENDP
```

Cette routine de test permet de vérifier le bon fonctionnement de l'interface et permet d'allumer et d'éteindre chaque relais. Partant de ce petit programme, bien des applications robotiques peuvent être envisagées, et à votre imagination de faire le reste... Par exemple décrocher le téléphone, puisque votre PSION sait composer un numéro, piloter une «tortue» électronique ou, pourquoi pas, un bras.

Pour ceux que les réalisations électroniques rebutent, pas de panique, vous pouvez vous procurer l'interface complète, montée et testée en envoyant un chèque de 690F (franco de port) à POCHE COMPUTERS.



Des accessoires pour votre PSION ?
comparez les prix (ttc)!!!

SSD RAM 128K : 735F SSD RAM 256K : 890F(nouveau !) SSD RAM 512K : 1490F
SSD RAM 1MB : 2920F : SSD FLASH 128K : 520F SSD FLASH 256K (nouveau !) : 720F
SSD FLASH 512K : 1220F SSD FLASH 1MB : 2090F SSD FLASH 2MB : 2890F
Lecteur de disquettes 1,44MB : 2790F Fax-modem : 2850F Emulateur minitel (câble + soft) : 1290F

ETC...

Faites confiance à un spécialiste des pockets de renommée nationale :
MICROCARDS - 42220 - BURDIGNES - FRANCE
Tél 77.39.68.13 Fax 77.39.19.60

FILE FOUND pour PSION series 3

Retrouver un fichier ? pas de problème avec ce programme de Gilles JEAN

```
REM Auteur ..... Gilles JEAN
REM Version ..... 1.0
APP Ff
        type 0
        icon»ff.pic»
ENDA
PROC START:
        global f$(128)
        global fich$(12)
        global volume$(32)
        global source$(128)
        global nbf%
        local ret%
        local vol$(3)
        local vol%
        local int%(8)
        lock on
        giprint «Gilles JEAN - Shareware diffusable v1.1F»
        trap close
        trap mkdir»loc::m:\opd» :REM Modif 1.1f
        trap delete»loc::m:\opd\ff.dat»
        trap create»loc::m:\opd\ff.dat»,a,a$
        vol$=»MAB»
        screen 38,6,2,2
        gat 0,2
        gborder 0,240,66
        do
                vol%=0 :nbf%=0
                dinit
                dtext «»,»Tapez tout ou partie du nom»,$2
                dtext «»,»du fichier à retrouver sans joker»,$202
                dedit fich$,»Chercher»
                if dialog
                        cls
                        do
                                vol%=vol%+1

source$=»loc::»+mid$(vol$,vol%,1)+»:\»

ret%=devinfo%:(source$,addr(int%(1)),addr
                                (volume$))
                                if ret%>=0
                                        media:
                                        setpath source$
                                        f$=dir$(source$)
                                        do
                                                file:
                                                f$=dir$(«»)
                                        until f$=»»
                                        while count
                                                first
                                                source$=a.a$+»\»
                                                erase
                                                setpath source$
                                                f$=dir$(source$)
                                                if len(f$)
                                                        do
                                                                file:
                                                        f$=dir$(«»)
                                                        until f$=»»
                                                endif
                                        endwh
                        endif
                until vol%=3
                        busy off
                        gat 0,77
                        gprintb «»,240,1
                        giprint gen$(nbf%,3)+» fichier(s) trouvé(s)»
                else
                        break
                endif
        until 0
        close
        trap delete»loc::m:\opd\ff.dat»
        f%=flgetat%:(f$)
        if (f% and $10)
                a.a$=f$
                append
        else
                print f$
                if loc(f$,fich$)
                        nbf%=nbf%+1
                        beep 5,500
                        giprint»Tapez sur une touche»
                        get
                        busy «Un instant SVP...»
                endif
        endif
ENDP
PROC MEDIA:
        gat 0,77
        gprintb upper$(source$)+» - «+upper$(volume$),240,1
        busy «Un instant SVP...»
ENDP
PROC devinfo%:(device$,pinfo%,pvol%)
        local exec%(10)
        local code%
        local rtn%
        local buffer$(64)
        local pbuffer%
        local dev$(129)
        local pdev%
        local i%
        code%=addr(exec%(1))
        pokew code%,$0Ab4
        pokew code%+2,$87cd
        pokew code%+4,$0272
        pokew code%+6,$c033
        pokeb code%+8,$cb
        pbuffer% = addr(buffer$)
        dev$ = device$+chr$(0)
        pdev% = addr(dev$)+1
        rtn% = usr(code%,0,pdev%,pbuffer%,0)
        if rtn% >= 0
                pokew pinfo%,peekw(pbuffer%)
                pokew pinfo%+2,peekw(pbuffer%+2)
                pokew pinfo%+4,peekw(pbuffer%+4)
                pokel pinfo%+6,peekl(pbuffer%+6)
                pokel pinfo%+10,peekl(pbuffer%+10)
                pokew pinfo%+14,peekw(pbuffer%+46)
                i%=0
                while peekb(pbuffer%+14+i%)<>0 and i%<=32
                        pokeb pvol%+1+i%,peekb(pbuffer%+14+i%)
                        i%=i%+1
                endwh
                pokeb pvol%,i%
        endif
        return rtn%
ENDP
```

FILE FOUND pour PSION series 3

```
proc flgetat%:(filenm$)
  local exec%(10)
  local code%
  local rtn%
  local pname%
  local nmcopy$(129)
  local buffer$(40)
  local pbuffer%
  code%=addr(exec%(1))
  pokew code%,$08b4
  pokew code%+2,$87cd
  pokew code%+4,$0272
  pokew code%+6,$c033
  pokeb code%+8,$cb
  pbuffer% = addr(buffer$)
  nmcopy$ = filenm$
  pname% = addr(nmcopy$)
  pokeb pname% + len(nmcopy$)+1,0
  rtn%=usr(code%,0,pname%+1,pbuffer%,0)
  if rtn% >= 0
        pbuffer% = pbuffer% + 2
        rtn% = peekw(pbuffer%)
  endif
  return rtn%
endp
```

BACKUP pour PSION series 3

Vous connaissez tous ce qu'est un Backup ? Il s'agit d'une sauvegarde complète de vos informations : Gilles JEAN nous a concocté ce programme.

```
APP Backup3
        type 0
        icon «backup3.pic»
ENDA
PROC START:
 global f$(128),source$(128),destin$(128)
 global sumoct&,nbf&,nbe&
 local g%,p%
 lock on
 giprint»Gilles JEAN - Shareware diffusable v1.1F»
 source$=»loc::m:\»
 destin$=»loc::b:\»
 do
        trap close
        trap mkdir»loc::m:\opd» :REM Modif 1.1F
        trap delete»loc::m:\opd\backup.dat»
        trap create»loc::m:\opd\backup.dat»,a,a$
        setpath source$
        sumoct&=0
        nbf&=0
        nbe&=0
        dinit»Backup III»
        dfile source$,»Source»,4
        dfile destin$,»Destination»,4
        if dialog
          if source$=destin$
                giprint»Source & destination
                  doivent être différent»
          else
                fond:
                f$=dir$(source$)
                do
                  file:
                  f$=dir$(«»)
                until f$=»»
                do
                  if count
                        first
                        source$=a.a$+»\»
                        erase
                        setpath source$
                        f$=dir$(source$)
                        if len(f$)
                          do
                                file:
                                f$=dir$(«»)
                          until f$=»»
                        endif
          else break
          endif
        until 0
        gcls
        gclock off
        trap close
        trap delete»loc::m:\opd\backup.dat»
        dinit»Informations Backup III»
        dtext «»,gen$(sumoct&,12)+»
          Octet(s) tranféré(s)»,2
        dtext «»,gen$(nbf&,12)+» Fi-
          chier(s) transféré(s)»,2
        dtext «»,gen$(nbe&,12)+» Fi-
          chier(s) non transféré(s)»,2
        dbuttons «Annuler»,-27
        dialog
          endif
        else
          lock off
          break
        endif
 until 0
ENDP
PROC FOND:
 gupdate off
 gstyle 0
 gat 0,0 :gborder $200,240,25
 gat 5,10 :gprintb «Source : «+source$,230,2
 gat 5,20 :gprintb «Destination : «+destin$,230,2
 gat 0,30 :gborder $200,115,15
 gat 8,41 :gprintb»Début : «,50,2
 gat 55,41 :gprintb right$(gen$(100+hour,3),2)+»:»
      +right$(gen$(100+minute,3),2)+»:»+right$
(gen$(100+second,3),2),52,1
 gat 125,30 :gborder $200,115,15
 gat 133,41 :gprintb»Heure :»,55,2
 gat 190,34 :gclock on,1 or $20
 gat 0,50
 gborder $203,240,30
 gupdate on
ENDP
PROC file:
 local dec%(6),int%(6),err%,oct&,f%
 f$=parse$(f$,source$,dec%())
 gat 4,72
 gstyle 9
 gprintb right$(f$,len(f$)-7),230,2
 f%=flgetat%:(f$)
 if (f% and $10)
                a.a$=f$
                append
 else
                trap
MKDIR destin$+mid$(f$,dec%(3)+1,dec%(4)
                -(dec%(3)+1))
                err%=flinfo%:(f$,addr(int%(1)))
                if err%>-1
                oct&=peekl(addr(int%(3)))
                trap
copy f$,destin$+mid$(f$,dec%(3)+1,128)
                if err
                        giprint errS(err)
                        nbe&=nbe&+1
                else sumoct&=sumoct&+oct&
                        nbf&=nbf&+1
                endif
                endif
 endif
ENDP
PROC flinfo%:(filenm$,pstatus%)
 local exec%(10)
 local code%
 local rtn%
 local pname%
 local nmcopy$(129)
 local buffer$(40)
 local pbuffer%
 local ptostat%
 code%=addr(exec%(1))
 pokew code%,$08b4
 pokew code%+2,$87cd
 pokew code%+4,$0272
 pokew code%+6,$c033
 pokeb code%+8,$cb
 pbuffer% = addr(buffer$)
 nmcopy$ = filenm$
 pname% = addr(nmcopy$)
 pokeb pname% + len(nmcopy$)+1,0
 rtn%=usr(code%,0,pname%+1,pbuffer%,0)
 ptostat% = pstatus%
 pokew ptostat%,peekw(pbuffer%)
 ptostat% = ptostat% + 2
 pbuffer% = pbuffer% + 2
 pokew ptostat%,peekw(pbuffer%)
 ptostat% = ptostat% + 2
 pbuffer% = pbuffer% + 2
 pokel ptostat%,peekl(pbuffer%)
 ptostat% = ptostat% + 4
 pbuffer% = pbuffer% + 4
 pokel ptostat%,peekl(pbuffer%)
 return rtn%
ENDP
proc flgetat%:(filenm$)
 local exec%(10)
 local code%
 local rtn%
 local pname%
 local nmcopy$(129)
 local buffer$(40)
 local pbuffer%
 code%=addr(exec%(1))
 pokew code%,$08b4
 pokew code%+2,$87cd
 pokew code%+4,$0272
 pokew code%+6,$c033
 pokeb code%+8,$cb
 pbuffer% = addr(buffer$)
 nmcopy$ = filenm$
 pname% = addr(nmcopy$)
 pokeb pname% + len(nmcopy$)+1,0
 rtn%=usr(code%,0,pname%+1,pbuffer%,0)
 if rtn% >= 0
        pbuffer% = pbuffer% + 2
        rtn% = peekw(pbuffer%)
 endif
 return rtn%
endp
```

RUBRIQUE TI-85

Quelques programmes pour TI-85
par Stéphane OUVETTE

-RACINE: Mise en fraction des racines carrées, sous forme formelle et simplifiée.
-ZREELSTG: Conversion des variables numériques en type "chaine de caractères" (REAL -> STRING)
-RACINEC: Programme identique en programme «racine», mais cette fois-ci pour les variables complexes.
-POLYNOME: Programme qui écrit sous forme formelle les polynômes factorisés automatiquement.

PROGRAMME RACINE

```
:ANS->A6:IF (fPART A6)= =0:Then:A6:Zréelstg:Return:End
:1->A7:If A6<0:-1->A7
:1->A8
:Lbl A1
:(A6*A8)²->A10:A8+1->A8:If (iPart A10)<>A10
:Goto A1
:A8-1->A8:1->A6
:Lbl A2
:A10/A6->A11:A6+1->A6
:If A11<>(iPart A11)
:Goto A2
:√A11->A9
:If A9<>(iPart A9)
:Goto A2
:A6-1->A6:A9*A7->A9
:gcd(abs A8,abs A9)->A10:A8/A10->A8:A9/A10->A9
:A9:ZréelStg:Ans->A10
:If A6<>1:Then:A6:ZréelStg:»(«+A10+»√»+Ans+»)»->A10:And
:If A8= =1:Then:A10:Return:Else:A8:ZréelStg:A10+»/»+Ans->A10:End
```

! Ce programme nécessite la présence du sous programme ZréelStg.
Exemples d'utilisation:

Entrées:	Affichages:
0,692820323028:Racine ---> :Ans	(2√3)/5
0,333333333:Racine:Ans --->	1/3
789/456:Racine:Ans -------->	263/152

Ce programme, à employer en sous-programme, permet l'écriture des valeurs exactes des racines carrées ou des fractions, en les convertissant en des variables de type «STRING».
Syntaxe générale = Ans : Racine : Ans
N.B: Si au bout d'une trentaine de seconde le résultat n'est pas affiché, stopper le programme, le chiffre n'est pas simplifiable.

PROGRAM: ZREELSTG

```
:ANS->A1
:abs round(A1,3)->A2
:1->A7:» «->STR
:If fPart A2<>0:-2->A7
:Repeat (A2<10^(A7-1))
:(mod(A2,10^A7)-mod(A2,10^(A7-1)))/10^(A7-1)->B1
:If (A7= =1) and (fPart A1<>0):».»+STR->STR
:sub(strng,B1+1,1)+STR->STR
:A7+1->A7:End
:If (iPart A1= =.0) and (A1<>0):»0.»+STR->STR
:If sign (A1)= =-1:»-»+STR->STR
:sub(STR,1,lngth STR-1)->STR:STR
```

Ce programme sert à convertir des variables de type REAL en type STRING. Il est indispensable au programme RACINE. Il est à noter que les chiffres convertis sont arrondis à 3 chiffres après la virgule.
Il nécessite la présence d'une chaine de caractères constante:
STRNG:»0123456789"

Syntaxe : Ans	: Zréelstg	: Ans
type real		type sting

strng «0123456789»

PROGRAM: RACINEC

```
:Ans->C1
:real C1:racine:»(«+Ans+»+i*»->C2
:imag C1:racine:C2+Ans+»)»->C2
```

Ce programme est l'équivalent des programmes «racine», mais pour les nombres complexes, qu'il utilise en sous-programme.
Le programme suivant permet l'écriture sous forme formelle de polynômes d'un degré ≥2 (mais <30).
Il nécessite les sous-programmes
-Z racinec -Racine -Zréelstg

PROGRAM: POLYNOME

```
:Input «Degré:»,D
:If D>2:Goto DEGRE3
:Disp «Ax²+Bx+C=0»
:Prompt A,B,C
:Lbl A2
:B²-4A*C->D
:Disp «Δ=B²-4A*C="
:D:racine:Disp Ans
:If D<0:Stop
:If D= =0:Disp «X=-B/2A=»,-B/2A▶Frac
:If D>0:Disp "X1=(-B+√D)/(2A)=",(-B+√D)/(2a)▶Frac
:Disp «X2=(-B-√D)/(2A)»,(-B-√D)/(2A)▶Frac
:If D>0:Then:Disp «  «:Outpt(8,&,»I  <>  <>  «):Outpt(8,2,sign
A):0utpt(8,ç,sign -A):Outpt(8,16,sign  A):End
:Stop
:Lbl DEGRE3
:Input «Coeff { }:»,CX
:» «->EQN2
:Lbl DEG
:» «->EQN:D+1->dimL CX
:For(I,1,(dimL CX)-1
:CX(I):racine
:EQN+Ans+»x»->EQN
:If I<dimL CX-1
:Then:(D+1)-I-:racine:EQN+»^»+Ans+»+»->EQN:End
:End
:EQN+»+»->EQN:CX(dimL CX):racine:EQN+Ans->EQN
:CILCD
:Outpt(1,1,EQN)
:If D= =2:Then:B->D:CX(1)->A:CX(2)->B:CX(3)->C:EQN2+»(«+EQN+»)»-
>eqn:Outpt(D+3,1,eqn):Pause :CILCD:Goto A2:End
:int ((lngth EQN)/21)+1->B
:St▶Eq(EQN,EQN):EQN
:ZRACINEV
:If R= =»N»
:Then:Eq▶St(EQN,EQN):EQN2+»(«+EQN+»)»-
>eqn:Outpt(B+6,1,eqn):Stop:End
:Outpt(B+2,1,»Racine evidente:»):Outpt(B+2,17,R)
:ST▶Eq(R,RE)
:0->A
:dimL CX->dimL CY
:FOR(I,1,D+1)
:RE*A->CY(I)
:CX(I)+CY(I)->A
:A->CX(I):End
:Outpt(B+3,1,CY)
:Outpt(B+5,1,CX)
:If CX(D+1)<>0
:Then:Eq▶St(EQN,EQN):EQN2+»(«+EQN+»)»->eqn:
Outpt(B+6,1,eqn):Stop:End
:D-1->D:
:EQN2+»(x-»+R+»)»->EQN2
:Eq▶St(EQN,EQN)
:Pause
:Goto DEG
```

PROGRAM: ZRACINEV

```
:E=EQN:»N»->R
:For(x,10,-10,-1)
:If E= =0
:Then:x:ZréelStg:Ans->R:Return:End
:End
:For(I,2,3)
:√I->x
:If E= =0
:Then:I:ZréelStg:»√»+Ans->R:Return:End
:End
:For(I,-10,10)
:If I= =0:1->I:I⁻¹->x
:If E= =0
:Then:I:ZréelStg:»(1/»+Ans+»)»->R:Return:End
:End
```

RUBRIQUE TI-85

PROGRAMME CVS

par Noël BONNET

Voici un programme qui s'utilise pour l'étude des «séries chronologiques». Il s'agit des «coéfficients des variations saisonnières».

1)Explications brèves du cours.

Dans les «séries chronologiques», beaucoup de données dépendent de leur date d'observation. Par exemple, une société vendra plus de maillots de bain l'été que l'hiver. On doit donc corriger ces observations dites «brutes» à l'aide de «coefficients de variations saisonnières» (CVS).

2)Calcul des CVS.

2.1Calcul de la moyenne générale.

C'est la somme des «y» (en restant dans les maillots de bain, ce serait le total des ventes) divisé par l'éffectif, c'est-à-dire le nombre de périodes données dans l'énoncé (souvent exprimé en mois, donc pour un an:12, pour deux ans:24, etc.).

$$Mgen=\frac{\text{Somme des «y»(sigma y)}}{\text{Effectif}}$$

2.2Calcul des moyennes mensuelles.

Par exemple pour une étude de deux ans, ce sont les moyennes de chaque mois (janvier, février, ..., novembre, décembre).

Pour janvier :

$$Mmen=\frac{\text{CA janvier 1° année + CA janvier 2° année}}{2}$$

Et ainsi de suite avec les autres mois.

2.3Calcul des CVS correspondants.

C'est la moyenne générale divisée par la moyenne mensuelle, il faut faire le calcul pour chaque mois.

$$CVS(janvier)=\frac{Mgen}{Mmen}$$

2.4Calcul des données corrigées.

Il s'obtient en divisant les observations «brutes» par les CVS correspondants.

$$\text{Donnée corrigée}=\frac{\text{Donnée brute}}{CVS}$$

2.5Chiffre d'affaires prévisionnel.

Il faut d'abord avoir fait une droite de MAYER ou les MOINDRE CARRES pour s'en servir. Cela permet à l'aide des CVS de trouver des valeurs plus réelles que celles sur la droite, ce à cause des variations en fonction des saisons.
CA brut = CA à partir de la droite * CVS

3)Fonctionnement du programme.

Tout se fait à partir d'un menu. Pour s'en servir, il suffit d'appuyer sur l'une des cinq touches de fonction (F1, F2, F3, F4, F5).
Il faut les éxecuter de gauche à droite, c'est très important car vous aurez besoin des premiers résultats pour faire les autres calculs.
Il faut faire le tableau en même temps car le programme suit l'ordre logique des opérations, et il faut concerver les résultats.
Donnez les valeurs demandées, le détail des calculs sera donné en même temps que le résultat.

3.1Explications des touches de fonctions F1 à F5.

F1 «Mgen»Calcul de la moyenne générale, puis retour au menu général.
F2 «Mmen»Calcul d'unemoyenne mensuelle, puis dans un autre menu, possibilité de faire une autre CVS ou de retourner au menu principal.
F3 «CVS» Calcul d'une CVS,puis dans un autre menu, possibilité de faire une autre CVS, ou de retourner au menu principal.
F4 «corrig»Calcul une donnée corrigée, puis dans un autre menu, possibilité de faire une autre donnée corrigée, ou de retourner au menu principal.
F5 «MORE»Permet de voir la suite du menu. Si on appuie à nouveau dessus on revient au menu principal.
F1 (après F5)«PREV»Calcul de chiffres d'affaire brut, puis dans un autre menu, possibilité de faire un autre CA brut, ou de retourner au menu principal.
F2 (après F5)«FIN»Pour sortir du programme.
Quand il est possible de choisir entre un autre calcul de la même nature et le retour au menu principal, le menu est le suivant :
F1 «AUTRE»On refait un calcul similaire.
F5 «FIN» On revient au menu principal.

PROGRAM: CVS

```
:CILCD
:Outpt(3,9,»RENTAL»
:Outpt(4,8,»PRESENTS»
:Outpt(6,10,»CVS»
:Pause
:Lbl DEB
:CILCD
:Disp «CORRECTION DES»
:Disp «VARIATIONS»
:Disp «SAISONNIERES»
:Lbl M1
:Menu(1,»Mgen»,MGEN,2,»Mmens»,
MMENS,3,»CVS»,CV,4,»Corrig»,COR,5,»
MORE»,M2
:Lbl M2
:Menu(1,»PREV»,PREV,2,»FIN»,FIN,5,
»MORE»,M1
:Lbl MGEN
:CILCD
:Disp «@y»
:Input Y
:Disp Y
:Pause
:Disp «Eff x(n)»
:Input E
:Y/E->MGE
:CILCD
:DISP «     @y «
:DISP «Mgen=————»
:DISP «     n «
:DISP MGE,MGE@Frac
:Pause
:Goto DEB
:Lbl MMENS
:CILCD
:Disp «Nbre d'années»
:Input A
:Lbl MMEN
:CILCD «@ par mois»
:Input Z
:Z/A->MME
:CILCD
:Disp «    @ par mois   «
:Disp «MME=————————»
:Disp «    Nbre d'années»
:Disp MME
:Menu (1,»AUTRE»,MMEN,5,»FIN»,DEB
:Lbl CV
:CILCD
:Disp «M par mois»
:Input M
:M/MGE->CVSS
:Disp «    M par mois»
:Disp «CVS=————————»
:Disp «    M général «
:Disp «»
:Disp CVSS
:Menu(1,»AUTRE»,CV,5,»FIN»,DEB
:Lbl COR
:CILCD
:Disp «donnée brute»
:Input AB
:Disp «CVS»
:Input CVSS
:AB/CVSS->DC
:Disp «            don.brute»
:Disp «don. corrig=————————»
:Disp «              CVS   «
:Disp «»
:Disp DC
:Menu(1,»AUTRE»,COR,5,»FIN»,DEB
:Lbl PREV
:CILCD
:Disp «CA à partir de la»
:Disp «droite»
:Input CAD
:Disp «CVS»
:Input CVSS
:CAD*CVSS->CAB
:CILCD
:Disp «CA BRUT=CA à partir»
:Disp «de la droite*CVS   «
:Disp «»
:Disp CAB
:Menu(1,»AUTRE»,PREV,5,»FIN»,DEB
:Lbl FIN
:CILCD
:Outpt(5,10,»FIN»
```

RUBRIQUE TI-85

Jeu MINES

par Jérôme FORESTIER

Ce jeu utilise1743 octets de programmes, 204 octets de réel, 1447 octets dans une matrice, et une chaîne de longueur variables. Il se compose de trois sous-programmes et d'un programme principal :
MINES : Programme principal (le plus gros).
MINES2 : Sous-programme pour remplir une case.
INKEY : Sous-programme attendant l'appui d'une touche et retournant sa valeur.
CLG : Sous-programme effacant l'écran graphique.

BUT DU JEU

Une fois les 4 programmes saisis, sans erreur, tapez MINES.

Le but du jeu est de parcourir un damier de 18X8 cases, dans lesquels sont cachés 18 mines, que vous ne voyez pas. Vous entrez en haut à gauche et devez sortir en bas à droite de l'écran.

En vous dirigeant dans les 8 directions (accessibles grâce aux touches 4 6 8 2 et 7 9 3 1 pour les diagonales), essayez de sortir du tableau. Le nombre de mines qui vous entourent est indiqué en bas de l'écran, par un nombre de points correspondant aux nombres de mines dans les 8 cases autour de vous. A chaque fois que vous vous déplacez, la case se colore en noir, pour vous indiquer sur quelles cases vous êtes déjà passé. Vous pouvez retourner sur une case noire.

Si vous sautez sur une mine le jeu s'arrête, et les mines cachées s'affichent, ainsi que votre score. Le score est calculé en fonction du nombre de cases parcouru avant la sortie. Essayez donc, pour un score maximal, d'"entourer" les mines. Si vous battez le record vous aurez la chance de faire la une des journaux...

Explication des programmes.

MINES : Le début du programme (LBL C) affiche le record, initialise l'écran graphique, place les mines dans le tableau A, et trace le damier.

Ensuite (LBL A), le programme détermine le nombre de mines autour de vous (variable B), effectue les tests de direction et de l'emplacement des mines, colorie la case. Si vous sautez sur une mine, leur emplacement est affiché (LBL B) et si votre score est supérieur au record, vous entrez votre nom.

INKEY : Sous-programme, qui grâce à une boucle WHILE attend la frappe d'une touche et retourne le code de la touche (variable KEY).

TRUCS : Il existe un cheat mode, à vous de le trouver.

Vous pouvez créer une image représentant le damier, faire des petits dessins dessus, et supprimer la ligne (prog MINES) :

For(A,1,18:Line(7A,1,7A,56):Line(1,7A,126,
7A:End

pour la remplacer par l'affichage de votre image: RcPic IMAGE (Si vous avez sauvé votre image avec ce nom).

Sachez que le jeu n'a pas de fin car à chaque fois que vous sortez, le jeu recommence, mais avec plus de mines.

IMPORTANT : Avant de lancer le jeu, vous devez taper les instructions suivantes sous peine de plantage :

0->RECM [ENTER]
(Initialisation du record).

« «->NOMM [ENTER]
(Initialisation du nom).

B O N D E M I N A G E ! ! !

```
PROGRAM:Clg
:FnOff
:ClDrw
PROGRAM:Inkey
:0->KEY
:While KEY==0
:GetKy->KEY
:End
PROGRAM:Mines
:0->S:18->M:O->R
:Lbl C
CILCD
:Disp «-Jérome FORESTIER-93-»
:Outpt(2,7,»M I N E S»
:Disp «»
:
:Disp «RECORD «+NOMM,RECM
:
:Clg
:0->xMin:126->xMax
:0->xScl:-5->yMin:57->yMax:0->yScl
:
:For(A,1,18:Line(7A,1,7A,56:Line(1,7A,126,
7A:END
:
:0->A
:(18,8->dim A
:
:For(B,1,M
:int (18rand+1->X
:int (8rand+1->Y
:If getKy==34:PtOn(7X-3,7Y-3
:Outpt(2,1,B
:1->A(X,Y
:End
:
:1->X:8->Y:0->A(1,8:0->A(18,1:0->A(1,7:0-
>A(2,7:0->A(2,8
:Mines2
:18->X:1->Y
:Mines2
:1->X:8->Y
:Line(4,-4,18,-4
:
:Lbl A
:DispG
:O->B
:If X<18 and Y<8:B+(A(X+1,Y+1)==1->B
:If Y<8:B+(A(X,Y+1)==1->B
:If X>1 and Y<8:B+(A(X-1,Y+1)==1->B
:If X>1:B+(A(X-1,Y)==1->B
:If X>1 and Y>1:B+(A(X-1,Y-1)==1->B
:If Y>1:B+(A(X,Y-1)==1->B
:If X<18 and Y>1:B+(A(X+1,Y-1)==1->B
:If X<18:B+(A(X+1,Y)==1->B
:For(C,1,B:PtOn(2+2C,-3:End
:For(C,B+1,8:PtOff(2+2C,-3:End
:PtOff((7X-3,7Y-3
:
:Inkey
:PtOn(7X-3,7Y-3
:KEY->K
:If K==0:Goto A
:X-(K==82->X
:X+(K==84->X
:Y+(K==73->Y
:Y-(K==93->Y
:Y+(K==72->Y:X-(K==72->X
:Y+(K==74->Y:X+(K==74->X
:Y-(K==94->Y:X+(K==94->X
:Y-(K==92->Y:X-(K==92->x
:If X>18:18->X
:If X<1:1->X
:If Y>8:8->Y
:If Y<1:1->Y
:S+(A(X,Y)==0->S
:
:If X==18 and Y==1Then:R+1->R:25+S(R>1-
>S:Goto B:End
:If 1==A(X,Y:Then:Circl(7X-3,7Y-3,6
:0->R:Goto B:End
:2->A(X,Y
:Mines2
:Goto A
:
:Lbl B
:For(X,1,18
:For(Y,1,8
:If 1==A(X,Y:PtOn(7X-3,7Y-3
:End:End:0->A:Pause
:CILCD
:Disp R,»Ton score: «,S:Pause
:If R>=1:Then:M+4(R>1->M:Goto C:End
:If S>=RECM:Then:Disp
«NOUVEAU RECORD!»,S:S->RECM:
InpSt«TON NOM»,NOMM:End
:Disp «RECORDS :»+NOMM,RECM
PROGRAM:Mines2
:For(ZZZ,7(X-1),(7X)-2
:Line(ZZZ+1,7Y-1,ZZZ+1,7(Y-1)+1
:End
```

RUBRIQUE TI-85

PROGRAMME SCROL (par Noël BONNET)

Ce programme permet de faire défiler un texte. Il suffit de changer le texte entre (« «) à la 6° ligne. Ce n'est pas un programme à part entière, il est plutot fait pour être incorporé dans d'autres programmes.

```
:CILCD
:0->A
:1->D
:20->F
:Lbl DEB
:Voici un exemple de texte déroulant, vous
pouvez mettre à cet endroit, ce que vous vou-
lez !!   «->TE
:lngth TE->L
:Lbl DED
:sub(TE,D,F->SCOS
:Outpt(8,1,SCOS
:D+1->D
:L-D->R
:if R>20
:Then
:Goto DED
:Else
:R+1->F
:sub(TE,D,F->SCOS
:Outpt(8,1,SCOS
:End
:Lbl FIN
:CILCD
:Outpt(3,9,»MADE»
:Outpt(4,10,»BY»
:Outpt'(5,8,»RENTAL»
```

PROGRAMME RESET (par Noël BONNET)

Ce programme ne sert pas à effacer tous les programmes que vous avez tapés. Au contraire, c'est un moyen de faire croire à une personne ne connaissant pas la machine, qu'en l'éxécutant, vous venez d'effacer tout ce que vous avez enregistré. Cela peut être utile pour certaines interros.... HUM HUM

Fonctionnement du programme
Tout d'abord, pour faire «plus vrai», quand vous aurez à vous servir du programme, évitez de passer par le menu des programmes «PRGM», il est préférable d'écrire en toutes lettres «RESET»
Ensuite il suffit de suivre les indications. Le plus important est de bien savoir jouer le comédien!!!

```
PROGRAM:RESET
:0->C
:CILCD
:Outpt(4,8,»RESET»
:Menu(1,»ALL»,AL,5,»PRGM»,AM
:Lbl AM
:1->C
:Lbl AL
:CILCD
:Disp «Are You Sure ?»
:Menu(1,»YES»,OUI,5,»NO»,NON
:Lbl NON
:CILCD
:Goto FIN
:Lbl OUI
:CILCD
:If C==1
:Then
:Disp «ALL PRGM RESET»
:Else
:Disp «ALL MEMORY RESET»
:End
:Goto FIN
:Lbl FIN
```

PUISSANCE 4 (par Jérôme FORESTIER)

But du jeu :
Il suffit d'essayer d'aligner 4 croix (x ou + suivant le joueur). Le premier joueur qui y arrive a gagné.

Appuyez sur le numéro de la colonne dans laquelle vous voulez jouer (1 à 7). Vous jouez à deux, l'un contre l'autre, le programme s'arrêtant seulement quand il n'y a plus de place pour jouer. Appuyez sur [2nd] pour recommencer une autre partie.

PUISSANCE 4 utilise 521 octets de programmes.124 octets de réel.156 octets de vecteurs.
Le jeu se compose de trois sous-programmes et d'un programme principal.
P4 Programme principal.
CLG Sous-programme éffacant l'écran graphique
INKEY Sous-programme attendant la frappe d'une touche et retournant sa valeur.
BITMAP Sous-programme utilisant le Range.

```
PROGRAM:Clg
:FnOff
:ClDrw
PROGRAM:Inkey
:0->KEY
:While KEY==0
:GetKy->KEY
:End
PROGRAM:P4
:Lbl ZZZ
:Clg:CILCD
:Bitmap
:1->J
:0->A
:For(A,0,7:Line(28+10A,1,28+10A,61:Line
(28,1+10A,98,1+10A:End
:7->dim A
:Lbl A:Inkey
:KEY->K
:If K==21:Goto ZZZ
:(K==92)+2(K==93)+3(K==94)+4(K==82)+5
(K==83)+6(K==84)+7(K==72->C
:If C>7 or C<1
:Goto A
:If 5<A(C
:Goto A
:28+10C->X:1+10A(C->Y
:If J==1
:Then
:Line(X-10,Y,X,Y+10:Line(X-10,Y+10,X,Y
:Else
:Line(X-5,Y,X-5,Y+10:Line(X,Y+5,X-10,Y+5
:End
:-J->J
:1+A(C->A(C
:Goto A
PROGRAM:bitmap
:1->xMin:127->xMax:0->xScl
:1->yMin:63->yMax:0->yScl
```

MISE AU POINT POUR PB1000

par V. LUGNIER

Voici un programme qui se propose de calculer le couple diaphragmme/mise au point idéal pour avoir une zone de profondeur de champs comprise entre 2 intervalles. Cela permet qu'un sujet ayant une certaine profondeur de champs soit entièrement net, tout en ayant la plus grande ouverture de diaphragme et un fond le plus flou possible.

```
10   CLS
20   E1=0:PM=0:DA=0:D1=0:DB=0:DX=0:DY=0:
DT=0:R2=SQR(2)
30   INPUT «FOCALE (mm)»;F:F=F/1000
40   INPUT «DEBUT NETTETE (m)»;D1
50   INPUT «FIN NETTETE (m)»;D2
60   E=0,00003:M1=100000
70   F1=F*F:W=E*F*D2:Z=D2*E:K=D2-D1
80   DG=(F1*(D2-D1))/(E*(2*D2*D1+F*(-D1-
D2))):DU=LOG D/(R2):DM=DU+1:P2=2*(D1*D2)/
(D1+D2)
90   FOR I=INT(DU)-1 TO INT(DM) STEP 1
100  D=R2^I
110  H=F1/(D*E)
130  A=2*H+K:B=-(2*H*F+2*K*F):C=K*(F1-
H*H)
140  DE=B*B-4*A*C:P1=(-B+SQR(DE))/(2*A)
150  P3=P1-F:H1=H*P1
160  DA=H1/(H+P3):DB=H1/(H-P3)
170  E1=ABS(DB-D2)
180  IF E1<M1 THEN GOTO 240
190  NEXT I
200  PRINT«D:»;ROUND(DX,-2);»P(m):»;
ROUND(PM,-3)
210  PRINT «ZONE NETTE (m):»;ROUND(DT,-
3);»—>»;ROUND(DY,-3):BEEP1
220  PRINT «DIAPH. THEO. «;ROUND(DG,-
5);»M.A.P. THEO. «;ROUND((P,-3)
230  END
240  M1=E1:DX=D:PM=P1:DT=DA:DY=DB:
GOTO 170
```

Explications

Le programme demande la focale (à donner en mm) puis la distance en mètres entre l'appareil photo et le 1r point qui doit être nette («Début netteté»), puis la distance en mètres séparant l'appareil photo du dernier point qui doit être nette.

Le programme calcule puis émet un bip sonore pour prévenir de la fin du calcul (qui peut durer + ou - longtemps) et affiche la zone de netteté réelle approchant au mieux la zone souhaitée (les distances données sont celles séparant l'appareil photo du premier plan net et du dernier.

Auparavant le programme donne le diaphragme à utiliser (D:) et la mise au point à régler en mètres ((P(m)) sur l'objectif.

Enfin il existe un couple diaphragme/mise au point théorique idéal correspondant exactement à la zone souhaitée donnée par « DIAPH. THEO. « et « M.A.P. THEO. «, mais la théorie n'est presque jamais praticable du fait que l'on ne peut régler le diaphragme que par valeurs entières.

Si l'on a la chance de pouvoir régler un diaphragme par demi valeur, voire par tiers de valeur, il suffit, *en ligne 90 de changer le STEP1 en STEP0,5* (demi valeur) ou en STEP1/3 (tiers de valeur), ainsi on aura un couple diaphragme/mise au point plus proche de l'idéal.

Ce programme convient pour les 24X36 en vue d'un agrandissement de 10 fois. Cela vient du cercle de confusion en 60 où E=0,00003. Pour un format moyen la valeur de E=0,000045 conviendra. Si l'on souhaite des agrandissements plus grands il faut utiliser E=0,000025 au moins. Notons que la valeur E=0,00003333 est celle utilisée par les fabricants.

Ce programme aurait pu être plus compact, mais moins rapide en remplaçant dans les équations comprises entre les lignes 90 et 180 les variables par leurs valeurs et ainsi supprimer la ligne 70, de même en 140 on aurait pu remplacer B*B par B^2, mais les élévations au carré sont plus lentes que la multiplication du chiffre par lui-même.

Pardonnez-moi l'inélégance de ce programme, mais au moins il marche. Je suis ouvert à toutes suggestions ou critiques.

INITIATION A L'ASSEMBLEUR 8086 (pour débutant)

par Thierry BODIN

Pourquoi programmer en Assembleur (ASM) alors que des langages dit de haut-niveaux existent (C,Pascal...) ? Pour plusieurs raisons: ce langage est le plus proche de la machine et a accès à toutes les fonctions de l'ordinateur. Il est pratiquement indispensable de recourir à l'assembleur lorsque une routine doit être effectué le plus rapidement possible (calcul, affichage, tri, ...). Bien sûr, les langages évolués sont eux-aussi rapides mais la plupart permettent d'inclure de l'assembleur pour accélérer les routines. De plus, l'ASM fournit un code compilé compacte. Par exemple le C créera un fichier d'une dizaine de kilo-octets pour afficher une simple phrase alors qu'en assembleur, la routine d'affichage prendra moins de 10 octets.

[INCONVENIENTS]

Quant aux inconvénients, les voici:

- Il faut plusieurs instructions pour obtenir quelque chose de correct d'où une approche et un apprentissage assez difficile .

L'affiche d'un message en C s'écrit:

```
PRINTF(«Je lis Poche Computers.»);
```

ou en Basic:

```
PRINT «Je lis Poche Computers.»
```

en assembleur, il faudra écrire:

```
        MOV   AH,9
        MOV   DX,OFFSET phrase
        INT   21h
phrase  DB    'Je lis Poche Computers.$'
```

(Nous expliquerons cet exemple plus loin)

- Il n'existe aucune routine qui permettent la manipulation des chaînes, la conversion d'un nombre décimal en hexadécimal et vice-versa, etc... Il faudra donc tout refaire, réinventer la roue en quelque sorte. (Voilà, j'ai perdu la moitié de mes lecteurs en écrivant cela !)

Il faut aussi signaler que ce langage comprend un jeu d'instructions réduit (un peu plus d'une centaine) dont certaines ont les mêmes fonctions.

Avant de se lancer dans l'apprentissage de l'ASM, une connaissance rudimentaire de la machine est nécessaire. Nous allons donc expliquer ce qu'est le binaire, l'hexadécimal, la mémoire, comment elle est structurée, les registres et les interruptions. Si vous avez un problème ou une question à poser, vous pouvez me contacter sur le 3615 RTEL en tapant *ECR puis SCARFY. N'oubliez pas de créer votre Boîte Aux Lettres (*BAL) en certifiant votre pseudonyme pour que je puisse vous répondre. A défaut, écrivez au journal qui transmettra.

[LE BINAIRE]

L'ordinateur comprend 2 états: 0 ou 1 l'équivalent électrique de éteint/allumé soit FAUX ou VRAI. On appelle cet état BIT qui regroupé par 8 donne un OCTET (ou BYTE), par 16 donne un MOT (ou WORD) ou par 32 donne un DOUBLE MOT (ouDOUBLE WORD) (Cf. Tableau A1).

[TABLEAU A1 - Nombres représentables]

Nbr BIT	NOM	Valeur stockable
1	BIT	0 ou 1
8	OCTET	0 à 255
		ou -128 a +127
16	MOT	0 à 65535
		ou -32768 à +32767
32	DOUBLE MOT	0 à 4294967295
		ou -2147483648 à
		+2147483647

Sur un OCTET, on peut stocker un nombre entre 0 et 255 soit 256 valeurs.

Parfait pour les nombres entiers mais pour les nombres négatifs, il faut trouver un moyen pour définir le signe. Vu qu'il n'y a que 2 signes possibles PLUS ou MOINS, ils peuvent être représentés sur un bit. Dans un ensemble de bits, c'est celui placé le plus à gauche qui sera le bit de signe. S'il est égal à 1 alors le nombre est négatif sinon il est positif.

Les 8 bits d'un octet sont numérotés comme ceci:

```
+-------------------------+
|      /— Octet —\ |
|      x x x x x x x x |
| Bit n° 7 6 5 4 3 2 1 0 |
+-------------------------+
```

Les premiers bits (0,1,..) sont appelés bits de poids faible et les derniers bits sont appelés bits de poids fort. Ils sont numérotés par ordre de puissance sachant que plus le bit est à gauche plus la puissance de 2 est grande.

Exemple: pour obtenir 16, il faudra calculer 2^4,

pour 44 = 2^5 + 2^3 + 2^2

[L'HEXADECIMAL]

La notation la plus fréquemment utilisée est l'hexadécimale ou base 16. Un chiffre hexadécimal comprend 16 éléments: les chiffres de 0 à 9 et pour les 6 éléments manquant, on a décidé de prendre les 6 premières lettres de l'alphabet soit A B C D E et F. Le A sera égal à 10, le B=11, le F=15 (cf. Tableau A2).

[TABLEAU A2 - Equivalence entre base 10, 2 et 16]

Décimal	Binaire	Hexadécimal
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
10	1010	A
15	1111	F
16	10000	10
60	111100	3C
105	1101001	69
255	11111111	FF

[LA MEMOIRE]

Il existe 2 types de mémoire: La RAM (Random Access Memory) et la ROM (Read Only Memory). Cette dernière ne peut être que lûe. C'est dans la ROM qu'est stocké, par exemple, le BIOS qui permet le démarrage de la machine. L'accès en lecture/écriture est possible en RAM.

La mémoire est composée de cellules numérotées de 8 bits soit 1 octet. Chaque numéro de cellule est appellé ADRESSE. La mémoire est divisible par zones de 65536 octets maximum ap-

INITIATION A L'ASSEMBLEUR 8086 (pour débutant)

pelées SEGMENTS et pour accéder à une cellule, il nous faudra donc 2 éléments: le numéro de SEGMENT et le déplacement (OFFSET) soit SEGMENT:OFFSET. Pourquoi une telle composition ? Tout simplement parce que le processeur INTEL 8088 fonctionne en 16 bits et sur ces bits, nous ne pouvons mettre que 65535 en nombre maximal, c'est à dire que nous pouvons lire 65536 octets (on compte à partir de 0) ou 64 Ko (1 Kilo-Octets = 1024 octets soit 64*1024= 65536). Sur les PC équipés de 640 Ko ou le Portfolio qui en a 128 Ko, il a fallu trouver un système pour accéder à cette mémoire supplémentaire. On a alors décidé que dans le segment, on mettrait un nombre qui, multiplié par 16, donnerait l'adresse absolue d'un emplacement mémoire. Par exemple si on veut accéder à l'octet n° 65537, on mettra dans le segment 65537/16 soit 4096 (1000h) et dans l'offset on mettra le reste soit 1. Donc on obtiendra 1000h:0001h. Si on ajoute 16 à l'offset et que l'on soustrait 16/16 soit 1 au segment, on obtiendra 4095:17 soit 0FFFh:0011h et ON POINTERA TOUJOURS SUR LE MEME OCTET ! (4095*16+17= 65537). Dans cette adresse, on pourra lire et/ou modifier son contenu (Consultez le tableau A3 sur la structure de la mémoire).

[TABLEAU A3 - STRUCTURE DE LA MEMOIRE]

Segment	Contenu
0000h	Vecteurs d'interruptions
0400h	Paramètres du BIOS
3F00h	Fin de la mémoire système(124K)
.	Zone pour les
.	extensions mémoire
.	jusqu'à 512 Ko
9F00h	
.	Inutilisé
.	
B000h	Mémoire vidéo MDA de 4 Ko
B100h	du Portfolio
.	Copie de la mémoire vidéo
C000h	(Non utilisé)
.	Mémoire Système ROM B (128Ko)
E000h	
.	Mémoire Système ROM A (128Ko)
FFFFh	

[LES REGISTRES]

Les registres sont des variables matérielles; le processeur y charge des données provenant de la mémoire, les traite à l'aide des instructions machine puis les renvoie en mémoire. Ils peuvent servir pour calculer, manipuler des chaînes ... Le processeur 8088 a 14 registres de 16 bits répartis en 4 groupes (voir tableau A4):

Registres de travail:

Ces 4 registres de 16 bits peuvent être décomposés en 2 registres de 8 bits. Par exemple si dans AX nous avons A13Eh (Le h indique une notation hexadécimale), AH est égal à A1h et AL est égal à 3Eh.

Si vous mettiez 0 dans AH, AX serait alors égal à 003Eh. Le H de AH signifie HIGH (FORT) et le L de AL signifie LOW (FAIBLE) C'est à dire octet de POIDS FORT et octet de POIDS FAIBLE.

Ces registres permettront le calcul et le stockage de résultats intermédiaires. Ce sont aussi les seuls à être divisibles en 2 registres de 8 bits.

Registres de segment:

Ils spécifient les segments utilisés. CS contient le numéro de segment où est situé le code compilé, DS et ES sont égal aux segments des données et SS contient le numéro de segment pour la pile.

Qu'est-ce que la pile ? C'est un tampon situé en mémoire où sont sauvés l'adresse de retour d'un programme lors d'un appel à un sous-programme, les paramètres transmis à une procédure (les langages évolués s'en servent beaucoup).

Registres d'offset:

Reliés avec les registres de segment, ces registres représentent des pointeurs. Par exemple, CS:IP pointe sur l'adresse de l'instruction en cours.

Registre flag:

Ce registre reflète le résultat d'opérations, averti en cas d'erreur... Sur les 16 bits que comporte ce registre, seulement 9 sont utilisés. Voir leur définition en tableau A5.

[TABLEAU A4 - LES REGISTRES]

REGISTRES DE TRAVAIL		
Nom	Composé de	Fonction
AX	AH et AL	Accumulateur
BX	BH et BL	Base
CX	CH et CL	Compteur
DX	DH et DL	Données

REGISTRES DE SEGMENT	
Nom	Fonction
DS	Seg. de données (DATA SEG)
ES	Extra segment
CS	Seg. de code (CODE SEG)
SS	Seg. de pile (STACK SEG)

REGISTRES D'OFFSET	
Nom	Fonction
SI	Source index
DI	Destination index
IP	Pointeur d'instruction
SP	Pointeur de pile
BP	Pointeur de base

REGISTRE FLAG

.	.	.	.	O	D	I	T	S	Z	.	A	.	P	.	C
15			11	9	8	7	6		4		2		0		
			10												

[TABLEAU A5 - Détails sur le registre FLAG]

Bit	Nom	Définition
0	CF (Carry flag)	Mis à 1 en cas d'erreur/reflète le résultat

INITIATION A L'ASSEMBLEUR 8086 (pour débutant)

```
|    |                    | d'un opération mathématique ou logique     |
| 2  | PF (Parity flag)   | 1 si le résultat d'une opération est paire |
|
| 4  | AF (Auxilliary flag) | Retenue auxiliaire                       |
| 6  | ZF (Zero flag)     | 1 si le résultat d'une opération est nul   |
|
| 7  | SF (Sign flag)     | 1 si nombre négatif                        |
| 8  | TF (Trap flag)     | 1 si mode pas à pas                        |
|    |                    | (utilisé par les debuggeur)                |
| 9  | IF (Interrupt flag) | 1 si appel des interruptions autorisé     |
| 10 | DF (Direction flag) | Si 1 DI et SI sont décrementés            |
|
|    |                    | Si 0 DI et SI sont incrémentés             |
|    |                    | (après instruction de traitement de données)|
| 11 | OF (Overflow flag) | Si 1: division par 0 ou résultat d'une     |
|
|    |                    | multiplication ou d'une division ne tenant |
|    |                    | pas sur un nombre de bits précis (déborde- |
|    |                    | ment ou Overflow). Aussi activé quand le   |
|    |                    | résultat de 2 nombres positifs additionés  |
|    |                    | est négatif                                |
+———¤———————————————¤———————————————+
```

[LES INTERRUPTIONS]

Qu'est-ce qu'une interruption ? C'est un ou plusieurs sous-programmes fournis par le DOS ou le BIOS. Parmi les 256 interruptions existantes, nous pouvons distinguer 2 catégories: les interruptions matérielles et les interruptions logiciels.

Les interruptions logicielles sont appelées par un programme pour utiliser toutes les fonctions qu'elles contiennent. Nous pouvons citer la numéro 16 (10h) qui s'occupe du vidéo. C'est elle qui affichera les phrases, les couleurs, qui passera en différents modes (graphiques ou textes). Il y a aussi la numéro 33 (21h) qui est la plus utilisée vu le nombre important de fonctions qu'elle propose. Elle permet d'ouvrir un fichier, de lire ou écrire dedans, ... Pour utiliser une des fonctions d'une interruption, on lui envoie un ou plusieurs paramètres par l'intermédiaire des registres.

Les interruptions matérielles sont appelées par la machine (l'interruption 8 (timer ou horloge) appelée 128 fois ou 1 fois par seconde selon la configuration choisie (18.2 fois par seconde sur un PC), l'interruption 9 qui gère le clavier et qui est appelée dès qu'une touche est tapée,...

[VOTRE PREMIER PROGRAMME]

Voici maintenant un programme simple que nous expliquerons plus en détails dans les lignes qui suivent. Vous pouvez l'écrire sous DEBUG mais nous vous conseillons fortement l'achat d'un compilateur tel que Turbo Assembler de Borland ou bien Microsoft Assembler. Pour compiler, tapez ce texte avec un éditeur puis, si vous avez TASM, exécutez sous DOS: TASM nomdufichier.ASM /M4 puis TLINK nomdufichier.obj /T. Vous aurez alors un fichier .COM.

Les apostrophes délimitent la phrase et lorsqu'ils sont répétés 2 fois, ils indiquent au compilateur que l'on veut en afficher un. 0Dh signifie que le curseur revient à gauche de l'écran, le 0Ah le fait descendre d'une ligne et le $ prévient l'interruption 21h que le message se termine ici. Le caractère ASCII 7 émettra un bip lors de l'exécution.

```
        CODE Segment     ;Nom du segment
        Assume cs:CODE   ;Previens le compilateur que CODE contient les
                         ;instructions
        Org     100h     ;Les programmes .COM commencent à l'offset 100h
LIGNE   equ     0Bh
COLONNE equ     20
Debut:
        mov     ax,0003  ;passage en mode texte
        int     10h      ;(80 colonnes, 25 lignes)
                         ;(Fonction 0, Sous Fonction 3/int 10h)
        mov     ah,2     ;Fct 2 /int 10h: Positionne le curseur à l'écran
        mov     bh,0     ;Page 0
        mov     dh,LIGNE ;N° de ligne
        mov     dl,COLONNE ;N° de colonne
        int     10h
        mov     ah,9     ;Fct 9/int 21h
        mov     dx,offset Message
        int     21h      ;Affiche un message
        mov     ax,0     ;Fct 0/int 16h
        int     16h      ;Attend une touche
        mov     ax,4C00h ;Fct 4Ch/int 21h Code d'erreur 0=OK
        int     21h      ;Retour au DOS
Message db      'Bienvenue dans le',7,' monde de l''Assembleur !',0Dh,0Ah,'$'
CODE Ends       ;Fin de segment
End Debut       ;Fin du programme
```

Nous vous conseillons de lire «ASSEMBLEUR facile» de Ph. Mercier édité par Marabout, ainsi que différents ouvrages décrivant la liste et les fonctions des interruptions (disponibles chez Sybex et Microsoft Press) et consulter les documents disponibles dans le domaine public tel que HELP PC ou bien INTERRUPTS LIST.

Expliquons maintenant l'exemple:

[EN-TETE]

Au début d'un programme assembleur, on doit donner un nom à chaque segment utilisé pour que le compilateur (et vous-même) s'y retrouve. Un programme d'extension .COM ne peut utiliser qu'UN SEUL SEGMENT d'où la nécessité de mettre le code et les données ensemble alors qu'un programme EXE est composé de plusieurs segments dont un réservé à la pile. Bien sûr, chaque segment ne fait pas 64k: il peut être de quelques octets (16) jusqu'à la limite des 64k, la taille d'un segment étant un multiple de 16.

Dans notre exemple le segment CODE contiendra le code-source compilé donc nous le précisons avec la DIRECTIVE D'ASSEMBLAGE ASSUME. Les fichiers COM doivent être exécutés à l'offset 256 (100h) d'un segment. Les informations contenues dans les 256 premiers octets étant utilisées par le DOS. Pour préciser alors au compilateur qu'il devra nous créer un fichier COM, on lui ajoute la directive ORG 100h (Origine).

[LES IDENTIFICATEURS]

Les identificateurs sont des noms définis par le programmeur et qui servent à indentifier une variable ou un label. Le premier caractère doit être une lettre et l'ensemble ne doit pas dépasser 80 caractères. Il ne faut pas non plus mettre d'expression mathématique du style TO-TO: ou TO+TO:.

Toujours dans notre exemple, DEBUT suivi de END DEBUT en fin de programme sont des LABELS qui indiquent au compilateur où se situe LE POINT D'ENTREE du programme soit

INITIATION A L'ASSEMBLEUR 8086 (pour débutant)

l'adresse de la première instruction exécutée. Pour un fichier COM, cette adresse est toujours fixe.

[LES CONSTANTES]

Les constantes sont des valeurs qui une fois définies par le programmeur ne peuvent être modifiées pendant l'exécution du programme. Pour les déclarer, on utilisera la pseudo-instruction EQU. (voir Tableau B1)

[TABLEAU B1]

```
+-------------------------------+
¦ Trois  EQU  10/5+1          ¦
¦ Cinq   EQU  Trois+2         ¦
¦ Huit   EQU  Trois+Cinq      ¦
¦                             ¦
¦ Vrai   EQU  1               ¦
¦ Faux   EQU  0               ¦
Ã-------------------------------Â
¦ Il n'y a pas de différence ¦
¦ entre les caractères       ¦
¦ minuscules et majuscules   ¦
+-------------------------------+
```

[LES CONSTANTES IMMEDIATES]

Ce sont des valeurs numériques exprimées sous différentes bases et définies pas le programmeur (par défaut: base décimale) et elles peuvent être aussi des caractères à condition d'être entre apostrophes. (Voir tableau B2). Attention: Toute valeur hexadécimale commençant par une lettre doit être précédée d'un 0 pour éviter la confusion avec un identificateur.

[TABLEAU B2]

Valeur	Exprimée en	De base
01001011b	Binaire	2
3123	Décimal	10
0EA3h	Hexadécimal	16
Valeur	Type	
'E'	Caractère	
'XYZD'	Chaîne de caractères	

[LES VARIABLES]

Les variables sont des cellules de la mémoire où l'on pourra lire ou écrire à volonté. Pour les déclarer, il suffit de leur donner un nom puis, selon la nature de la variable, vous ajoutez 'DB' (Define Byte) pour les variables contenant un ou plusieurs éléments tenant sur 1 octet, 'DW' (Define Word) pour les variables tenant sur 1 mot (16 bits - les 8 bits de poids faible sont stockés en mémoire avant les 8 bits de poids fort) ou 'DD' (Define Double word) pour les variables 32 bits (Double Mot). DD sert essentiellement à stocker des adresses (Segment:Offset). Le mot de poids faible, comme pour DW, est stocké avant le mot de poids fort. Si la valeur à déclarer n'est pas fixée, mettez un point d'interrogation. DUP permet de DUPliquer les éléments entre parenthèses. (Voir tableau B3)

[TABLEAU B3]

```
¦ Types de déclarations:                ¦
¦                                       ¦
¦ Initialisation:                       ¦
¦ Nomvar  db    0                       ¦
¦ Nomvar2 dw    0                       ¦
¦ Nomvar3 dd    0                       ¦
¦ Déclaration non initialisée:          ¦
¦ Nomvar  db    ?                       ¦
¦ Nomvar2 dw    ?                       ¦
¦ Nomvar3 dd    ?                       ¦
¦ Plusieurs éléments:                   ¦
¦ Nomvar  db    1,4,6                   ¦
¦ Nomvar2 dw    55626,10,4532           ¦
¦ Nomvar3 dd    642567,5,0A23E5FC1h     ¦
¦ Tableau non initialisé de 15 éléments: ¦
¦ Nomvar  db    15 dup (?)              ¦
¦ Nomvar2 dw    15 dup (?)              ¦
¦ Nomvar3 dd    15 dup (?)              ¦
¦ Table:                                ¦
¦ Nomvar  db    8 dup (1,3,5)           ¦
¦ Caractère:                            ¦
¦ Nomvar  db    'F'                     ¦
¦ Chaîne:                               ¦
¦ Nomvar  db    'Réponse Micro'         ¦
```

[MODES D'ADRESSAGE]

Nous allons voir ici comment manipuler les données situées en mémoire en utilisant l'instruction MOV. Bien-sûr, il existe d'autres

instructions pour manipuler les données mais MOV est l'instruction la plus utilisée dans un programme assembleur. Avant d'aller plus loin dans l'explication de notre exemple, arrêtons-nous un peu pour bien comprendre les différents modes d'adressages:

Dans notre premier listing, vous pouvez lire MOV AX,0003 qui peut se résumer à MOV AX,3. C'est ce qu'on appelle l'ADRESSAGE IMMEDIAT:

L'instruction MOV va ranger 3 dans le registre AX (AH=0 et AL=3).

REGISTRE <- VALEUR IMMEDIATE

Quant à l'ADRESSAGE DIRECT, il consiste à lire une valeur située dans une variable. Par exemple, nous avons une variable de nom EP1 de type MOT située dans le même segment que le code machine. Pour la lire, il faudra préciser le segment où elle se trouve ainsi que son nom (voir tableau B4).

REGISTRE <- MEMOIRE

[TABLEAU B4]

```
¦ EP1   dw    3A2Bh       ¦
¦                         ¦
¦     MOV   AX,CS:[EP1]   ¦
Ã-------------------------Â
¦     MOV   AX,CS         ¦
¦     MOV   DS,AX         ¦
¦                         ¦
¦     MOV   AX,[EP1]      ¦
¦ équivaut à:             ¦
¦     MOV   AX,DS:[EP1]   ¦
¦ et à:                   ¦
¦     MOV   AX,EP1        ¦
```

Pour simplifier l'accès à ces données, il a été décidé que DS serait le SEGMENT DE DONNEES à condition qu'il pointe sur le bon segment. L'erreur à ne pas commettre est de vouloir adresser un registre n'étant pas du même type que la variable. MOV CL,DS:[EP1] est incorrect car CL est un registre 8 bits alors que la variable EP1 est déclarée 16 bits d'où erreur annoncée par l'assembleur. L'ADRESSAGE BASÉ permet de lire une donnée par l'intermédiaire du registre BX qui représente l'OFFSET de la donnée (voir tableau B5). REGISTRE <- CS:[BX] ou

INITIATION A L'ASSEMBLEUR 8086 (pour débutant)

DS:[BX] (adressage basé sur CS ou DS)
Le mode d'adressage REGISTRE <- SS:[BP] (adressage basé sur SS) est identique au précédent.

```
      [TABLEAU B5]
¦     MOV    BX,128                          ¦
¦     MOV    CX,[BX]                         ¦
¦     ou                                     ¦
¦     MOV    CL,[BX]                         ¦
¦ Peut aussi s'utiliser comme ceci:          ¦
¦ EP1   DB     14                            ¦
¦ EP2   DW     1210                          ¦
¦                                            ¦
¦     MOV    BX,OFFSET EP1  ; OFFSET demande à l'assem-
¦
¦     MOV    AL,[BX]        ; bleur de mettre l'adresse ¦
¦     MOV    BX,OFFSET EP2  ; de EP1 dans BX            ¦
¦     MOV    DX,[EP2]                        ¦
¦ S'utilise aussi sous cette forme:          ¦
¦     MOV    BX,OFFSET EP2                   ¦
¦     MOV    BX,[BX]        ; BX prendra la valeur      ¦
¦                    ; contenue dans DS:BX   ¦
¦ Pour chaque ligne d'un commentaire, il faut mettre un  ¦
¦ point-virgule.                             ¦
```

L'ADRESSAGE INDEXÉ
Cet adressage facilite la manipulation des chaînes de caractères ou la
localisation d'un élément dans un tableau. DS est la base utilisée mais
ES peut être utilisé lors des traitements des chaînes de caractères
(voir tableau B6).
REGISTRE <- DS:[SI] ou DS:[DI] (adressage basé sur DS ou bien ES)

```
      [TABLEAU B6]
¦     MOV    SI,OFFSET EP2                   ¦
¦     MOV    DI,OFFSET EP1                   ¦
¦     MOV    AX,[SI]        ; AX prendra la valeur  ¦
¦                    ; d'EP2 soit 1210 (4BAh) ¦
¦     MOV    [DI],AL        ; EP1 sera égal à BAh   ¦
```

Très vite, on s'apercevra de la limite de ces adressages. On connaissait déjà l'adressage via les registres ou via une adresse mais il n'était pas encore assez pratique. Nous allons alors utiliser un autre type d'adressage qui permettra de traiter des données en se positionnant dessus grâce à plusieurs registres qui , additionnés, donneront alors l'adresse de l'information. C'est le cas de l'ADRESSAGE INDEXÉ ET BASÉ. En voici le principe: (tableau B7).

```
      [TABLEAU B7]
¦ TABLEAU_1     DB     43,12,67,55,24,91,35        ¦
¦     MOV    BX,OFFSET TABLEAU_1                  ¦
¦          ; BX pointe sur le 1er élément         ¦
¦          ; de la table TABLEAU_1                ¦
¦     MOV    SI,4                            ¦
¦     MOV    AL,[BX][SI]                     ¦
¦          ; S'écrit aussi MOV AL,[BX+SI]        ¦
¦          ; BX additionné à SI pointe alors     ¦
¦          ; sur le 5ème élément du tableau      ¦
¦          ; (Le 1er élément étant au déplacement ¦
¦          ; 0). AL prend alors la valeur 24     ¦
¦                                                ¦
¦  REGISTRE <- DS:[BX+SI] ou DS:[BX+DI] (ES peut être  ¦
¦          utilisé à la place de DS)             ¦
```

Si on utilise le registre BP à la place de BX, SS est le registre de segment par défaut car BP sert principalement à lire les informations contenues dans la pile. Pour comprendre le fonctionnement de la PILE, imaginez plusieurs assiettes empilées. Quand on rajoute une assiette, on la place au-dessus. Dans la même idée, si on en prend une, on prend celle posée en dernière, soit la plus au-dessus sinon il y a de forte chance d'avoir à utiliser la pelle et le balais ! La pile d'un programme fonctionne exactement pareil:
Quand on place une information sur la pile (instruction PUSH que nous expliquerons plus tard), par exemple le nombre 3342h, cette information se trouve au sommet de la pile. C'est pour cela que lors de la lecture de la pile (instruction POP), le premier nombre à sortir sera 3342h (tableau B8).

```
      [TABLEAU B8]
¦     MOV    AX,1223                         ¦
¦     PUSH   AX                              ¦
¦     MOV    AX,4342                         ¦
¦     PUSH   AX                              ¦
¦     MOV    AX,4512                         ¦
¦     PUSH   AX                              ¦
¦     MOV    AX,9876                         ¦
¦     PUSH   AX                              ¦
¦ ; A chaque PUSH, SP est décrémenté de 2     ¦
¦ ; et la donnée est sauvée dans SS:[SP]      ¦
¦                                             ¦
¦ La pile contient alors:                     ¦
¦ 1er élément 9876                            ¦
¦ 2ème        4512                            ¦
¦ 3ème        4342                            ¦
¦ 4ème        1223                            ¦
¦                                             ¦
¦ Si SP pointait sur l'adresse 100 avant l'empilage, ¦
¦ il pointera maintenant sur l'adresse 92 et on      ¦
¦ obtiendra en mémoire:                       ¦
¦                                             ¦
¦ Adresse:  92   94   96   98   100           ¦
¦ Contenu: 9876 4512 4342 1223   -            ¦
```

Il y a encore mieux: On peut utiliser les registres ET une adresse pour préciser les coordonnées d'une donnée. Par exemple dans BX vous avez le déplacement et pour gagner du temps, on ne fera pas MOV SI,OFFSET TABLEAU
mais MOV AX,[OFFSET TABLEAU+BX] ou bien MOV AX,[BX+OFFSET TABLEAU].
De même, MOV CL,[SI+OFFSET TABLEAU] ou encore MOV DH,[DI+2] fonctionne parfaitement. Le tableau B9 résume tout cela.

```
      [TABLEAU B9]
¦ TABLEAU dw     9,1,8,2,7,3,6,4               ¦
¦                                              ¦
¦ MOV    SI,4                                  ¦
¦ MOV    BX,3                                  ¦
¦ MOV    DI,2                                  ¦
¦ MOV    AX,[SI+OFFSET TABLEAU]                ¦
```

FIN DE LA PREMIERE PARTIE :
La suite dans le prochain numéro de POCHE COMPUTERS

GESTIONNAIRE DE MACROS COMMANDES

par JM POULOT

Ce logiciel intéressera plus d'un possesseur de PC compatible ; il a l'avantage de tourner également sur les Pockets tels que Sharp PC3000, HP95-100,Olivetti, Pocket PC etc... Vous pouvez obtenir la version d'évaluation (déjà très fonctionnelle) directement chez l'auteur pour 20F en timbres ou la version complète pour 250F)

1. Historique

L'idée d'un programme de macro-commandes est née en utilisant AutoCad. Ce logiciel, performant au niveau DAO, est un peu pénible quant à l'accès de ses commandes. Ne disposant pas de tablette graphique car peu adaptée à mon travail,il fallait déplacer la souris et cliquer sur les options et menus, (peu pratique sur un grand écran) ou taper les commandes au clavier.

Je voulais obtenir une fonction AutoCad à partir d'une touche (ou combinaison) en réduisant ainsi les temps morts (non productifs) et conserver les yeux sur mon travail.

La première étape consista donc à associer une suite de caractères à une touche de fonction. Les douze touches étant utilisées, il fallait imaginer des combinaisons avec les touches ALT et CTRL. Ces nouvelles possibilités apportèrent déjà une souplesse intéressante mais insuffisante. Les touches SHIFT ont été aussi ajoutées pour multiplier les combinaisons

Mais les séquences de caractères étant figées, il apparut la nécessité d'y inclure des fonctions : simulation de touches, signal sonore, test des boutons de la souris, branchements conditionnels et répétition des macros, sous-programmes , gestion des plusieurs fichiers ...

Toutes ces fonctions débouchèrent sur la création d'un pseudo langage et d'un compilateur. La version de cette disquette est limitée non dans ses fonctions mais dans le nombre exécutées (20). Après avoir épuiser ce nombre, il faudra désinstaller puis réinstaller MACRO pour utiliser vos macros.

Pour créer des macros-commandes, il faut donc écrire un fichier source, le compiler, et éxécuter le gestionnaire avec pour paramètre le fichier objet issu de la compilation.

2. Structure d'une macro-commande

A l'aide du compilateur (qui est aussi un éditeur de texte simple) il faut décrire les macros. Si cet éditeur ne convient pas, il est tout à fait possible d'en utiliser un autre.

Une macro commence TOUJOURS par le caractère] seul sur une ligne. Ensuite vient le nom de la touche (ou combinaison) d'appel de la macro toujours sur une ligne. Puis la séquence de caractères, de codes ou fonctions représentant la macro à éxécuter. La fin de la description est déterminée par le caractère] seul sur une ligne.

Exemple de macro qui, sous DOS, effectue un DIR par la touche F1 :

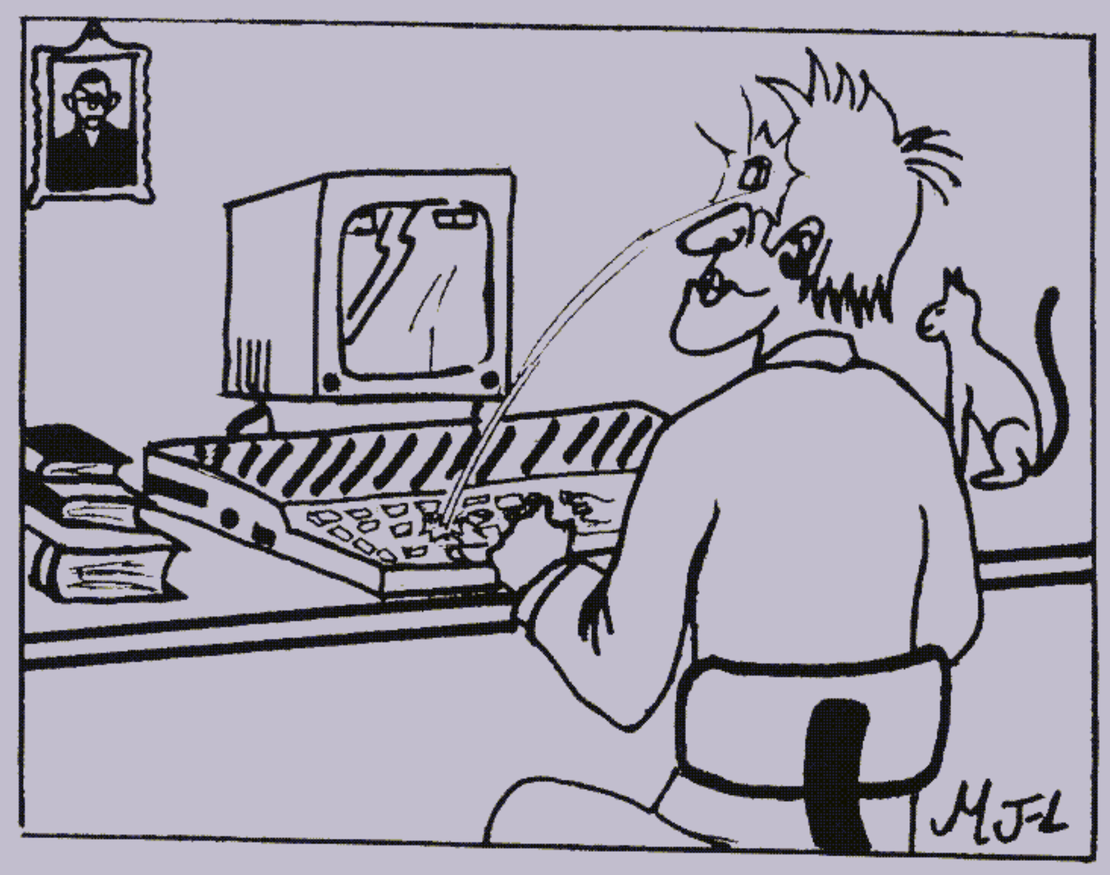
]	caractère d'ouverture de macro
F1	nom de la touche d'appel
dir	séquence de caractère
#cr	simulation de la touche ENTREE
]	fin de la macro

Dans le fichier texte appelé SOURCE, on peut décrire toute une série de macros relatives à un programme particulier. Le compilateur vérifiera la synthaxe des commandes ainsi que les noms des touches d'appel.

3. Touches d'appel

Les noms possibles sont :

F1 à F12
ALT-F1 à ALT-F12



CTRL-F1 à CTRL-F12
SHFTD-F1 à SHFTD-F12
SHFTG-F1 à SHFTG-F12
CTRL-SHFTD-F1 à CTRL-SHFTD-F12
CTRL-SHFTG-F1 à CTRL-SHFTG-F12
ALT-SHFTD-F1 à ALT-SHFTD-F12
ALT-SHFTG-F1 à ALT-SHFTD-F12
ALT-A à ALT-Z
CTRL-A à CTRL-Z
SHFTD-A à SHFTD-Z
SHFTG-A à SHFTG-Z
CTRL-SHFTD-A à CTRL-SHFTD-Z
CTRL-SHFTG-A à CTRL-SHFTG-Z
ALT-SHFTD-A à ALT-SHFTD-Z
ALT-SHFTG-A à ALT-SHFTG-Z
ALT-0 à ALT-9
CTRL-0 à CTRL-9
SHFTD-0 à SHFTD-9
SHFTG-0 à SHFTG-9
CTRL-SHFTD-0 à CTRL-SHFTD-9
CTRL-SHFTG-0 à CTRL-SHFTG-9
ALT-SHFTD-0 à ALT-SHFTD-9
ALT-SHFTG-0 à ALT-SHFTG-9
ALT-ESC,CTRL-ESC,SHFTD-ESC,SHFTG-ESC
CTRL-SHFTD-ESC,CTRL-SHFTG-ESC
ALT-SHFTD-ESC,ALT-SHFTG-ESC

Les touches numériques sont les chiffres situés en haut du clavier et non pas ceux du pavé numériques. L'expérience m'a prouvé que ces touches étaient plus accessibles en cours de travail.

Ces combinaisons (424 possibilités !) de touches peuvent déclencher une macro dès lors que le gestionnaire a été installé AVANT l'application.

ATTENTION ! Si une macro est définie sur une touche (ou com-

binaison) et que le programme (application) l'utilise déjà, c'est la macro qui est prioritaire.
Il est néanmoins possible de simuler les touches de fonctions (F1 à F12).

4. Codes et fonctions

Dans une macro, tout caractère est considéré comme code clavier à transmettre au programme. Si une fonction doit être entrée, il faut la faire précéder par le caractère #. (voir ci-dessus l'exemple).

Description des fonctions # :

#PA suspend l'éxécution de la macro jusqu'à l'appui de ENTREE
#RP reprend l'éxécution de la macro depuis son début
#DEB redéfinit le début de la macro sur cette ligne
#FIN provoque la fin de la macro
#RTS provoque la fin d'un sous-programme (retour à la macro appelante)
#CR simule la touche ENTREE
#TB TAB
#DL DEL
#F1... F1 ...
#F12 F12
#UP CURSEUR HAUT
#DN CURSEUR BAS
#CG CURSEUR GAUCHE
#CD CURSEUR DROITE
#PU PAGE UP
#PD PAGE DOWN
#HM HOME
#ED END
#ES ESCAPE
#HR envoie l'heure courante sur 2 caractères (00-23)
#MN les minutes (00-59)
#SD les secondes (00-59)
#JR le jour (01-31)
#MS le mois (01-12)
#AN l'année (00-99)
#MSG suspend l'éxécution jusqu'au clic du bouton Gauche (souris)
#MSD bouton Droite
#MSM bouton Milieu
#MSN attend relâchement d'un quelconque bouton de la souris
#MS? attend l'appui de n'importe quel bouton de la souris
#SI MSD teste le bouton Droit, exécute la ligne suivante si vrai
#SI MSM Milieu
#SI MSG Gauche
#AFN x,y affiche le message (ligne suivante) col x ligne y
#MSX x déplace le curseur de la souris col x
#MSY y ligne y
#COD x place le caractère défini par le code ascii x (0-255)
#SUB x appel du sous-programme (PDEF) x (0-255)
#BIP x émet un bip sonore 0:pause, 1:grave à 32:aigu
#TMP x règle le tempo de BIP en 1/18ème sec (1-200)

5. Particularités

Pour définir une macro, on déclare la touche d'appel. Pour un sous-programme, on inscrit PDEF et le numéro de la routine.

Exemple

Définition du sous-programme n°10

```
[
PDEF 10
#TMP 10
#BIP 20
]
```

Définition de la macro appelée par F1

```
[
F1
#SUB 10
DIR /P
#CR
#SUB 10
]
```

Les sous-programmes doivent être déclarés AVANT les macros qui les utilisent. Les sous-programmes peuvent aussi appeler d'autres sous-programmes dans la limite de 16 imbrications.

Une macro particulière est la macro automatique.
Définie par le label AUTO, elle peut être déclenchée de 2 façons

- à intervalles de temps réguliers (ex sauvegardes, alertes ..)
- immédiatement au lancement du gestionnaire (ex initialisation)
- depuis une macro

Les modes d'exécution de AUTO sont déterminés par les paramètres transmis au gestionnaire lors de l'installation.

Lors de l'éxécution d'une macro, le tour de l'écran prend une couleur particulière et renseigne sur l'état de la macro.
Par défaut les couleurs sont :

- bleu : macro en cours
- rouge : macro en attente d'un évennement
- bleu clignotant : macro AUTO dans 5 secondes
- bleu fixe : macro AUTO en cours
- vert : macro AUTO en cours superposant une autre (à la fin de AUTO, la macro interrompue reprend)

6. Installation

On suppose l'exemple suivant :
dans le fichier source nommé TEST.DEF on inscrit

```
[
F1
cls
#cr
]
```

après compilation (touche F9 dans CMAC) on obtient un fichier objet
du nom de TEST.@@@
faire sous dos MACRO TEST puis ENTREE (la touche !)
un message indique le déroulement de l'installation
et à chaque appui de F1 l'ordre DOS CLS s'éxécute

GESTIONNAIRE DE MACROS COMMANDES

Pour désinstaller le gestionnaire, faire MACRO puis ENTREE

Soit en règle générale:

- lancer MACRO avec le fichier objet (installation des macros)
- lancer l'application et travaill
- à la fin de l'application, lancer MACRO (désinstallation)

Il est très pratique d'opérer avec les procédures batch pour lancer une application avec ses macros.

Autres:

On peut à tout moment interrompre le déroulement d'une macro en appuyant simultatnément les touches CTRL et ALT.

Lors de l'utilisation d'un éditeur de texte autre que CMAC, pour compiler le fichier, faire CMAC [fichier] /A.

7. Paramètres à l'installation

/N désactive le tour d'écran lors de l'éxécution d'une macro les couleurs peuvent être modifiées avec COULEUR

/F accélère les traitements, certains programmes n'acceptent pas ce mode (MicroSoft pour ne pas les citer)

/G les macros ne seront actives qu'en mode graphique (attention avec les drivers ADI !!)

/I déclenche immédiatement la macro AUTO (si elle existe...)

/x x=valeur d'intervalle de temps de 1 à 59 minutes pour AUTO en mode turbo cette valeur dépend de la vitesse de la machine

8. Généralités

Pour utiliser au mieux MACRO et CMAC, il est préférable de placer ceux-ci dans la racine du disque (ou un répertoire réservé).

Avec PATH=C:\ par exemple, il peut être appelé depuis n'importe quel répertoire.

Ensuite, pour profiter des macros dans une application, créer un fichier batch

comme suit :

```
rem installation des macros en mémoire
macro [fichier] [/n] [/x] [/g] [/f]
rem test si bien déroulé
if errorlevel 1 goto erreur
ici lancer l'application
truc.exe
rem désinstaller les macros
macro
fin du fichier batch
goto fin
:erreur
pause
:fin
```

9. Evolution prochaine de MACRO :

- la gestion de la souris clics et déplacements
- la simulation de toutes les touches et combinaisons
- compilation sur 2 passes
- l'appel inter-macro (ex: la macro F1 appel la macro CTRL-Q)
- variables numériques et chaines de caractères
- boucles d'itération FOR, WHILE, REPEAT UNTIL
- ouverture de fenêtre pour choix d'options multiples
- sauvegarde d'écrans textes et graphiques
- gestion de la mémoire étendue
- mode d'ENREGISTREMENT dynamique
- changement de fichier de macros sans désinstaller
- inhibition des macros sur commande directe et évènementielle
- éxecution automatique séquentielle contrôlée (temps, evènements)

Autres produits :

- menu AutoCad : interface permettant la gestion des fichiers AutoCad

ouverture, traçage, impression, macro-commandes etc...

- lanceur d'application garphique sous Dos par icônes

9. Autre

A l'installation de MACRO, un contrôle de CRC est effectué. Si un virus ouune modification du code intervient, la machine se bloque , un message indiquant la nature du défaut.

10. Contribution et enregistrement

La version de MACRO présente sur cette disquette est limitée dans le nombre de macros exécutées (20) dans une session. Il faut désinstaller puis réinstaller MACRO pour fonctionner à nouveau.

Pour obtenir la version complète de MACRO et CMAC, une contribution de 250 FF est demandée. Cette participation donne accès à des mises à jour régulières car ce produit évolue toujours.

En outre, un manuel d'utilisation est fourni et décrit toutes les ressources cachées de MACRO qui ne sont pas décrites ici.

Utilisez ce programme en respectant l'esprit SHAREWARE, vous pouvez copier ce logiciel à condition de fournir toujours ce fichier d'accompagnement avec les programmes sur le même support.

L'auteur se réserve le droit d'accorder ou non l'enregistrement d'un utilisateur et décline toute responsabilité des conséquences qui résultent de l'utilisation de MACRO et CMAC.

Pour tous renseignements (ou suggestions), contactez :

Jean Marc POULOT
1, rue principale
39350 DAMMARTIN

ACCUMULATEURS ULTRA PUISSANTS
(1200mA) de type R6 AA
tension nominale 1,2V, tension maximale 1,5V
rechargeables 1000 fois
avec chargeur du commerce
se mettent en lieu et place des piles
sont écologiques car sans cadnium
APPLICATIONS :
micros de poche - photo
aéromodèlisme - radio - etc...
PRIX :
1 à 2 unités : 46F ttc unitaire
3 à 5 unités : 43F ttc unitaire
6 à 10 unités : 40F ttc unitaire
au delà : 36F ttc unitaire
Disponibles sur stock
MICROCARDS - 42220 - BURDIGNES
tél 77.39.68.13 fax 77.39.19.60

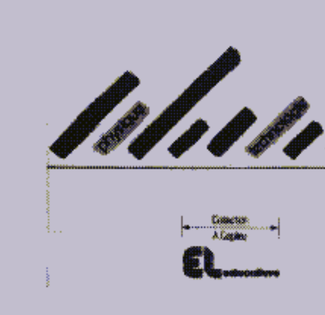
LES LIVRES

LES LIVRES DU BAC

Deux nouveaux titres viennent enrichir la collection Calculatrices Efficaces de Dunod. Il s'agit de titres destinés aux futurs bacheliers souhaitant exploiter efficacement mais simplement leurs outils de calcul. «HP-48 pour le Bac» et «TI-82, TI-85 pour le Bac» sont des livres uniques en leur genre puisqu'ils comprennent à la fois : «ce qu'il faut savoir» de votre machine, un résumé du cours de mathématiques, et surtout, de très nombreux sujets du bac corrigés «calculatrice en main». «HP-48 pour le Bac» et «TI-82, TI-85 pour le Bac» par Loïc Fieux et Henri Lemberg, édités par Dunod, ces deux livres seront disponibles dès la fin du mois d'avril 94.

Le MEMO FORMULAIRE

A posséder absolument !
Plus de 300 pages de formules en tout genres.

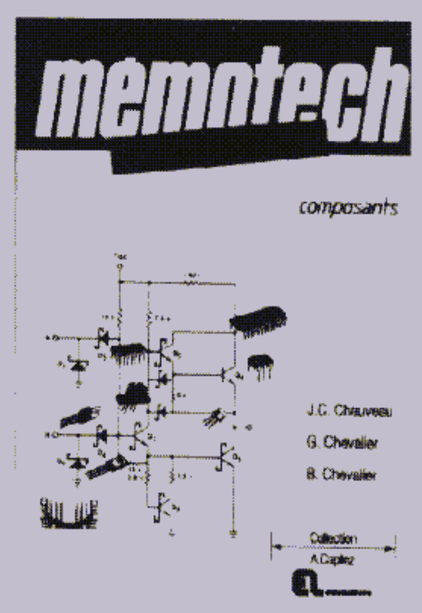


Saluons l'heureuse initiative des éditions CASTEILLA qui ont édité ce livre dont les qualificatifs manquent pour le décrire ! Que vous soyez étudiant, chercheur, ou tout simplement curieux de toutes sciences, vous trouverez là réponse à tous vos problèmes de formules. Jugez plutôt de l'éventail proposé :

- mathématiques
- physique
- électricité - électronique
- chimie
- biologie
- technologie
- unités
- vie pratique

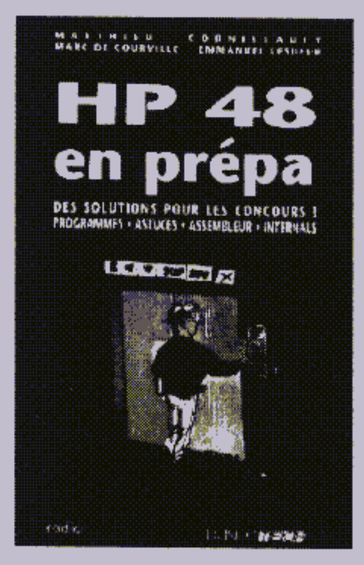
Vous recherchez la formule de la racine cubique ? ; celle de la période d'un réacteur ? ; celle d'un moment polaire d'une couronne ? ; du potentiel électrique ? ; de la permutation en chimie organique ? ; de l'abaque des impédances en automatisme ? etc... Pas de problème, consultez le MEMO FORMULAIRE.
L'ensemble est particulièrement bien structuré et accompagné de multiples dessins, croquis ou tableaux. La recherche s'effectue très rapidement et on se prend vite au jeu de la connaissance empirique. **Un livre à mettre entre toutes les mains** et à conserver sur sa table de chevet !!! De plus, le prix est modique.

Aux éditions CASTEILLA également, le **mémotech électronique** pour tout savoir sur les composants électroniques. Depuis les normes internationales, jusqu'à la liste des constructeurs et organismes, en passant par la description détaillée de la plupart des gammes de composants : circuits logiques - analogiques - optoélectroniques - composants passifs et actifs - semiconducteurs
transformateurs et relais - connecteurs
- protections électroniques diverses.
TOUT Y EST !!! le livre de référence de tout lycéen ou étudiant en électronique ou du passionné.



HP 48 en prépa

Ce livre, aux éditions DUNOD, sorti voici un an et demi est plus que jamais d'actualité.
L'étudiant y trouvera de très nombreux programmes tels que :
*Matrices et polynômes symboliques
*Racines des polynômes
*Inverse de matrice
Ainsi que les aides aux concours : Mines - Ponts - X - Ulm etc...



HP48 - Pour le Bac

(Henri LEMBERG et Loïc FIEUX), aux éditions DUNOD
250 pages - 125F -

Vous avez l'ambition de réussir votre bac haut la main et vous disposez de la "Rolls" des calculatrices graphiques, une HP48G ou GX... Bravo, Il ne vous manquait plus que ce livre pour disposer de la panoplie complète vous permettant de venir à bout des sujets de maths les plus récalcitrants ! Jugez plutôt :
-Prise en main de la HP48G-GX
-Résumé des cours de maths de terminale
-Recueil d'annales du BAC de 1989à 93

HP 48 - Permis de conduire

(Loïc FIEUX) aux éditions DUNOD
192 pages - 98F

Déroutante pour le débutant, passionnante pour l'initié, telle est la HP48. Destiné aux profanes de la notation polonaise inversée, ce nouveau petit livre d'initiation à la HP 48SX/GX s'est promis de vous faire brûler les étapes, car, quoi de mieux qu'une acquisition rapide et aisée de "l'esprit HP" pour exploiter rapidement et facilement sa calculatrice ?

ANIMATION pour GRAPHIQUE CASIO

(troisième et dernière partie)

par David WINTER

Dans cette dernière partie, nous étudierons la partie graphique. Rappelons que la partie graphique correspond à ce que l'on appelle la RAM VIDEO ; c'est à dire une zone en RAM dans laquelle sont sauvegardées les données de l'écran graphique. Pour pouvoir comprendre ce qui va suivre, il est impératif d'avoir compris l'article précédent au niveau des accès en RAM, et surtout du codage d'une mémoire. Malheureusement, je n'ai pu trouver les adresses graphiques des autres machines que la 8500... elle est égale à Z (4919).

1) La ram vidéo :

Les dimensions de l'écran de la calculatrice sont 96x64. Or, la ram vidéo se code dans 96 mémoires situées entre Z (4919) et Z (5014), en ce qui concerne la 8500. Cela signifie que chaque mémoire représente un carré de 8x8 pixels sur l'écran.

ATTENTION: Seule la Casio 7500G possède un codage différent, se faisant ligne par ligne. Dans le cas de cette machine, chaque mémoire codera pour une ligne de 64 pixels. Mais le principe de codage est le même, bien sûr.

Voici le plan de la ram vidéo de la 8500. Pour les autres machines exceptée la 7500, il suffira de remplacer les adresses.

64

	491	492	492	492	492	492	492	492	492	492	492	493
	493											494
	494											495
	495											496
	496											497
	497											499
	499											500
0	500	500	500	500	500	500	500	501	501	501	501	501
	0											96

2) Codage d'une mémoire vidéo :

Il vous est demandé avant tout de bien connaître le principe de codage d'une mémoire. Référez-vous à l'article de POCHE COMPUTERS N°2 si vous ne savez pas comment opérer. Chaque mémoire représente, nous le savons, 8x8 pixels, et elle se code sur 16 chiffres dont deux pour l'exposant, un pour les signes de l'exposant et de la mantisse. La méthode est la même que pour les caractères texte :

- décomposer les 13 chiffres pour obtenir une séquence de 13 chiffres.
- ajouter l'exposant et les deux chiffres codant celui-ci.
- placer la virgule entre les 2 premiers chiffres.
- en fonction de la valeur du chiffre codant l'exposant et la mantisse, placer les deux signes et modifier l'exposant si son signe est négatif.
- décomposer la séquence finale en une somme de deux nombres de façon à ce que la machine prenne en compte tous les chiffres.

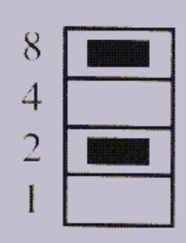
Voici à présent le schéma d'une mémoire :

8-4-2	Exp2	13	11	9	7	5	3	Sign
8-4-2	Exp1	12	10	8	6	4	2	1
	1	2	3	4	5	6	7	8

Chaque partie de cette grille est une colonne de 4 pixels. Un chiffre va donc représenter 4 pixels ; son codage se présente comme suit :

Il faut calculer la somme totale des pixels à éteindre. Pour cela, on part de 0 et à chaque fois que, en se déplaçant sur la grille dans une colonne, on a un pixel éteint, on ajoute à la somme totale le chiffre qui lui correspond. L'écriture en ram vidéo est telle que l'on ne peut mettre qu'un chiffre compris entre 0 et 9. Il sera donc impossible de coder des séquences correspondant à un chiffre supérieur à 9. Mais on pourra compenser avec l'instruction PLOT pour allumer quelques pixels...

Par exemple, nous avons une colonne de 4 pixels qui représente la séquence suivante :



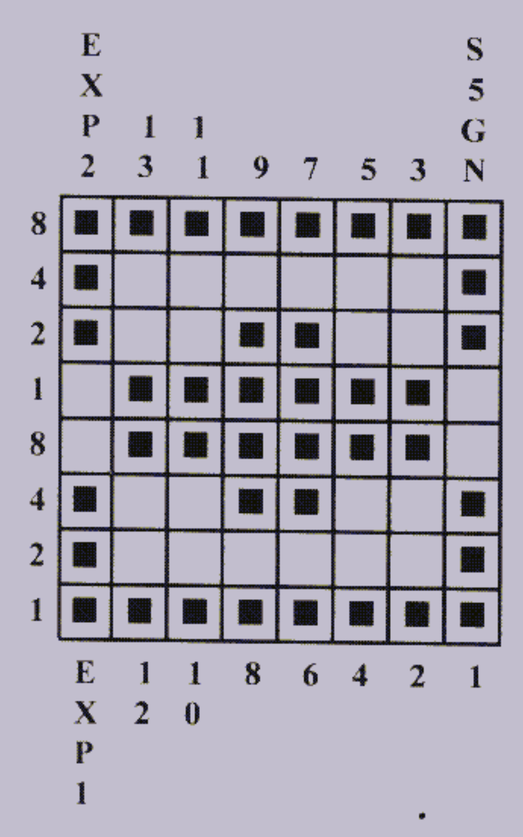
Le chiffre codant pour cette séquence sera donc égal à 5. En effet, le pixel 4 est éteint ainsi que le pixel 1. Les pixels 2 et 8 sont allumés.
Nous avons alors 1+4=5

Le codage de l'exposant et de la mantisse reste le même avec toujours 4 possibilités :

Chiffre	Mantisse	Exposant
0	+	+
1	+	-
5	-	+
6	-	-

3) Exemple de codage d'une mémoire pour un bloc de 8x8 pixels :

Voici d'abord la grille comportant les pixels du bloc à coder.
Au dessus et en dessous de la grille, figure la place du chiffre dans la séquence.



ANIMATION pour GRAPHIQUE CASIO

La méthode de codage est la suivante :
- Commencer par trouver les 13 chiffres ; ici, la séquence est : 8666624246666.
- Placer la virgule entre les deux premiers ; on obtient 8,666624246666
- Chercher ensuite les deux chiffres codant l'exposant ; ici, le premier est égal à 8 et le second à 1 : l'exposant est donc égal à 81. On a alors 8,666624246666E81
- Calculer la valeur du chiffre de la colonne "signe" et en déduire les signes de l'exposant et de la mantisse : égal à 1, ils sont tous les deux positifs. On a donc +8,666624246666E+81.

Or, la machine prend en compte les dix premiers chiffres, donc nous allons devoir faire une somme de deux nombres. Par exemple, nous pouvons mettre : 8.66662424E81 + 6666E69.
Attention : Ne jamais oublier le cas où le signe de l'exposant est négatif ; à ce moment, il faut retirer 100 à sa valeur absolue. Se référer à l'exemple donné dans POCHE COMPUTERS N°2.

4) Conclusion

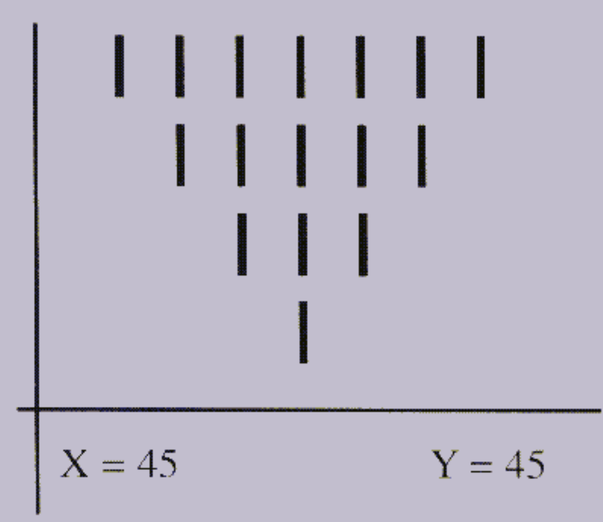
Vous avez maintenant tout ce qu'il faut pour faire des animations avec votre machine. Mais il existera des cas où il sera impossible d'obtenir certains dessins, à cause du défaut de codage...
Cependant, on pourra toujours faire des petits programmes, par exemple de dessin avec sauvegarde d'écrans. Vous trouverez ci-après un programme de sauvegarde d'écrans. Sa syntaxe est correcte, mais vous pourrez remarquer qu'il fonctionne rarement car certaines mémoires sont codées, à la base, de telle façon qu'il est impossible de les copier dans d'autres mémoires. C'est dans ce cas que certains chiffres la codant sont supérieurs à 9, donc compris entre A et F en hexadécimal.

```
" 1 => save"
" 2 => load"
? -> A                    ;demande saisie de A
A<1 => 1 -> A             ;vérifications de la valeur de A
A>2 => 2 ->A
"Ecran N° (0 -> 7)"
7 ->B                     ;demande saisie de B
B<0 => 0 -> B             ;vérifications de la valeur de B
B>7 => 7 -> B
97B -> B                  ;adresse de sauvegarde
4919 -> C                 ;adresse graphique de la 8500
Lbl 0                     ;label pour la boucle de copie
A=1 => Z [C ->Z [B]       ;sauve écran si demandé
A=2 => Z [B ->Z [C]       ;charge écran si demandé
Isz B                     ;on passe aux mémoires suivantes
Isz C
B<97 => Goto 0            ;continue la copie si B<97
"READY..."
```

Jeu des allumettes (dit jeu de MARIENBAL) pour Casio FX

(par Emmanuel MEUNIER)

Ce jeu se joue à deux. Son but est de prendre un certain nombre d'allumettes par ligne. Chacun, à son tour, peut piocher autant d'allumettes qu'il désire, mais sur une seule ligne seulement ! Le perdant est bien sûr celui qui doit prendre la dernière allumette.



Pour commencer une partie, allez sur la ligne du bas de l'écran.

```
Mcl
Range 0,94,0,-12,50,0
Lbl 0
-10-->G
Lbl 3
Plot 55-A+G,5+A
Plot 55-A+G,11+B
Line
G+10-->G
2A+G=>Goto 3
A+10-->A
B+10-->B
B≠40=>Goto 0
45 --> X  Y
Lbl 1
Plot  X,Y
Y= 0 =>Goto 2
10 Int((Y-5)/10)+8-->D
10 Int((X+5)/10)+8-->I
Plot 49-D,D
Plot I,D
Line
Goto 1
Lbl 2
"FIN"
```

RUBRIQUE GRAPHIQUE CASIO

Voici une version améliorée et allégée du fameux jeu "Démineur" sur Casio 8800

par Olivier ARTRES

```
Defm 67
Range .125,11.875,0,-.875,6.875,0.5 -->
A
3-->A
3-->V
Lbl 0
A <= 6.875 => Graph Y=A
A >11.875 =>Goto3
Plot A,0
Plot A,6.875
Line
Lbl 3
Int (2ran#) --> Z[V]
Lsz A
Lsz V
V≤ 66 => Goto 0
1 --> A~B
0 --> Z[12]~ Z[13]
Lbl 2
K= 0 => A<11 => Lsz A
Int (2Ran#) >K
K= 1 => B<6 => Lsz B
0 --> Z [A+11(B-1)]
A=11 =>B=6 => Goto 5
Goto 2
Lbl 5
1 -->A ~B
0 -->V
Lbl 1
0 -->E
"Nb de mines"
A>1 =>Z[A-1+11(B-1)] -->E
A<11 =>Z[A+1}11(B-1)] +E -->E
B<6=>Z[A+11B]+E-->E
B>1=>Z[A+11(B-2)]+E-->E
A>1=>B<6=>Z[A-1+11B]+E-->E
A>1=>B>1=>Z[A-1+11(B-2)]+E-->E
A<11=>B<6=>Z[A+1+11B]+E-->E
A<11=>B>1=>Z[A+1+11(B-2)]+E-->E
E▲
Plot A-.1,B+.1
Plot X,B-.1
Plot X,B-.1
Line
Plot A,Y
Plot X,B+.1
Line
Plot A+.1,Y
Plot X,B-.1
Line
Plot A,B▲
A-->C
B-->D
X<A=>Dsz A
X>C=>Lsz A
Y<B=>Dsz B
Y>D=>Lsz B
A<1=>C-->A
A>11=>C-->A
B<1=>D-->B
B>11=>D-->B
A=11=>B=6=>Goto 7
Z[A+11(B-1)]=1=>Goto 8
Goto 1
Lbl 7
«Gagné»▲
Goto 4
Lbl 8
«Perdu»▲
Lbl 4
Lsz V
V=66=>Goto 9
Z[V]=0=>Goto 4
V-11(Int((V-1)/11))-.5-->A
Int((V-1)/11)+.5-->B
Plot A,B
Plot A+1, B+1
Line
Plot X,B
Plot A,B+1
Line
Goto 4
Lbl 9
Defm 0
Graph Y=0▲
```

GOLF 3D (par Emmanuel MEUNIER)

REGLES DU JEU

Appréciation de la distance :Plus le drapeau est petit, plus l'écran est profond.

Appréciation du vent : Le drapeau donne la direction et la force du vent.

Appréciation de la pente :

Saisie des données :Tirs de longues distances (supérieurs à 15 mètres). Quand le curseur est placé en bas de l'écran (47,0), déplacez-le pour indiquer la force (en Y) et la direction du coup (en X), car le vent dévie la balle. Ensuite, après avoir tapé sur EXE, le curseur se situe dans le schéma de la pente, en haut à gauche. Placez-vous sur le demi-cercle pour indiquer l'angle de frappe. ATTENTION, c'est l'anle formé avec le sol, et non avec l'horizontale. L'angle doit être positif. Tirs de courte distance (Putt).

Indication : Il y a un P dans le soleil. Il ne faut saisir que la force (X est inefficace). ATTENTION, la pente intervient car la balle roule !!

RESULTAT : Si vous avez fait moins de coups que le «par», c'est gagné !!

PROGRAMME GOLF 3D

```
MCL:»===GOLF 3D===»
Int 80 Ram II-40->K
Tan K->K
47->D
Int 22 Ram II+15->P
Int 18 RamAM II-8->I
«PAR:»:Int(1.5+PAbs KI
/60->Z▲
Lbl 9
Range 0,94,0,-8,54,0,
0,Π,Π/5
0->A~C
Rad
Lbl 0
Plot 24-24A/P,P-A
Plot 69+24A/P,P-A:Line
Isz B
A+(Bx^y 2.2)/3->A
B<6=>Goto0
Lbl 1
Plot C,0:Plot 24+C/2,P:Line
C+13.2->C
C<95=>Goto 1
Lbl 3
300^(P-1)->H
Plot D,P:Plot D,P+H:Line
Plot D+10 IP^-1,P+0.43 H+10Abs IP^-1:Line:
Plot D,P+0.75H:Line
Graph (X,Y)=(4 Cos T+85,4 Sin T+45)
Graph(X,Y)=(4Cos(T+Π)+85,4
Sin(T+Π)+45)
Graph (X,Y)=(8 Cos(T-Π/2)+10,8 Sin(T-
Π/2)+42)
Plot 5,42:Plot 10,42:Line
Plot 10+8Cos Tan^-1 K:Line
P>15=>Goto 5
Plot 84,43:Plot 84,47:Line
Plot 86,47:Line:Plot 86,45:Line
Plot 84,45:Line
Lbl 15
Deg:Plot 47,0▲
Y->F
X-47->L:Isz W
P>15=>Plot 10,42▲
P>15=>Cos^-1((X-10)/8->Q
Abs(2F²(CosQ)²(K-Tan Q)/10->S
L+FI/7->U
U<0=> -I->I
P=<15=>Int(F-8K)->P
P>15=>Int(√((P-S)²+U²)->P
P>35=>»TROP FORT !!»▲
P>35=>Goto 5
S>P=>-K->K
P=<8=>Goto 7
K(P-S)/P->K
« «:» «:» «:» «:» «:» «:» «
Goto 9
Lbl 7
W=<Z=>»GAGNE !!»▲
w>Z=>»PERDU !!»▲
«JOUEUR :»
W▲
«BONUS :»
Z-W▲
```

PETITES ANNONCES GRATUITES...

(maxi 150 caractères ou espaces)

Les petites annonces sont publiées sous l'entière responsabilité de leur auteur. Nous nous réservons le droit de ne pas publier certaines petites annonces n'ayant aucune relation directe ou indirecte avec les micros de poche ou les calculatrices.

TI82 cherche cordon transfert + programme pour relier PC à TI82 ou 85 échange programme TI82 - Bruno ROUSSEAU - 10 impasse du sud 94120 FONTENAY - 48.72.69.77

O.MONACHON 4 rue delagrange 75015 Paris (1) 42.50.64.89 Vds PORFOLIO 128K INTERFACE PARALLELE + CABLE + TURBO PASCAL + T.FORTH = 1750 F à débattre

Vds organiser CASIO SF9300, jamais servi, encore sous plastique, cause double emploi. 1000 F au lieu de 2360 F aujourd'hui! TEL: (1) 47.80.87.14

Vds PSION serie 3 256K avec TABLEUR et DOC TEL: (99) 56.39.63 Heure repas.

Vds PSION series 3 256KO 1200 F TEL: 94 03.40.10 - (soirée) - Frederic, 83 TOULON

Vds TI74 Basicalc + 3 modules: MATH, RAM 8K et PASCAL 800 F TBE William au (16) 42.81.82.77

Vds Sharp PC 3000 1MO SS garantie 9 mois 2000 F tél : 21 58.74.20

O.MONACHON 4.R. DELAGRANGE 75015 PARIS TEL: (1) 42.50.64.89 HP48SX + carte HP41 + KIT PC + GOODIES DISKS + LIVRES + BATT. RECHARG. total: 3500F, vendu 2000F à débattre.

Vds programme de gestion de comptes + échéanciers pour scharp EL-93OO .J'etudie vos besoins spécifiques. tel: 60.16.87.96 Alain FOURNIER (Paris)

Vds Porfolio 128K + carte RAM128 + cartes ROM BASIC et MATH + ADAPTEUR SECTEUR: 2000 F à débattre. TEL: 64.62.29.86 - Benoit LOEILLOT

Vds Porfolio 128 + carte 64 KO 600 F. Vds TURBO FORTH 350 F TEL: (21) 58.74.20

Vends CASIO 8500G avec 3 MANUELS DE PROGRAMMATION TBE: 100 F TEL:(1) 39.66.01.13 Paul.

Vds pour sharp PC-1350 ou pour PC-1450P: impri. table traçante CE-140P (7 couleurs) état neuf PRIX : 700 F. Appeler le soir en semaine : 88.60.53.75

Vds pour PB-1000: LECT. DISQ. MD-100 (nombreux programmes) état neuf 2000 F + Impri. thermique FP-40 500 F. Appeler le soir en semaine. TEL: 88.60.53.75

Vds HP95LX 512KO + Connection TV PACK + Documentation en anglais + programmes palmtop suscaiber Mr GARCON-DUFOUR TEL: 64.21.40.92 26 Rue de la couture 77500 CHELLES. Prix: 1800 Frs

Vds organiseur SHARP IQ-8100M (64K) + carte BASIC/MATHS + TRADUCTEUR anglais + tableur + RAM32K + imprimante + housse cuir: valeur environ égale à 5800 F / Prix: 2000 F / jean: 45.76.51.70

Vds PSION Series 3a 512KO SSD 128K + cable impr. + Sacoche luxe + gestion bank + jeux. sous garantie. Valeur: +5000 Frs. Vendu: 3900 Frs. TEL: 47.32.07.41 TARTU.

Vds PSION 3 128K GARANTIE 10/94 1000F + traducteur 300 Frs. TEL: 38.72.66.24 AP. 18 H

Cherche modem + coupleur acoustique + soft + hard (ROM) pour porfolio ; Vends Casio 850P WE: SUCHET Dominique 70200 Linexert TEL: 84.63.60.44

Vds TI85 état neuf + interface PC et MAC le tout sous garantie (oct 95) valeur 1300F vendu 980 F port compris (métropole) TEL: 29.35.13.44 Beaume. Epinal.

Vends HP 48GX sous garantie 1500 F + secrets de la HP48 230 Frs, et câble PC, le tout 1900 FRS. Ecrire Joan MOREAU 20 RUE CHAMBERT 37000 tours.

Vds organiseur casio SF-R20 256KO tableur 3D, GAR 6 MOIS: 2000 Frs SHARP PC1600 + table traçante A4 4 couleurs lecteur disque + 32KO RAM: 1500 Frs à debatre, TEL: 20.91.07.81

Vds SHARP PC-1500 + table traçante; canon X07; extensions et documentation système. TEL: (1) 60.02.58.80

Cherche lecteur de disquettes ou autres extentions pour CANON X-07. LAMBERT Pascal 31 rue du BRABANT 37100 TOURS.

Vds HP 48GX sous garantie + câble liaison PC + nombreux programmes 2400 Frs port compris, à discuter Raphael RACHE. 63, chemin des HUTINS 1232 CONFIGNON (Suisse)

Echange sur HP48 SX/GX progs, jeux. Possède 1M de progs et de jeux. LEMARCHAND Stéphane 1 rue des Fermes 76310 Ste Adresse.

Vds programme de transfert + câble RS232 pour SHARP PCE5OO + interface série. TEL: 72.24.17.97 (Après 19 H).

Achète HPI6C un bon prix si elle est complète et en TBE. Francois au 47.71.70.46

Vds PSION LZ64 + étui + adaptateur secteur + tableur + data pack 16K + soft + liaison mac + manuels - 1200 Frs ARGELLY TEL + FAX: 19.41.21.94.34.329 (Le soir).

PETITES ANNONCES GRATUITES...

Vends HP28S + manuels.
Bon état : 450F Sandra SPIESSER
16 rue cigognes
67330 NEUWILLER/SAVERNE
TEL: 88.70.04.39

Vds casio FX 8000 G état neuf + doc VF + Livre «jeux et graphisme sur casio FX» : 450 Frs. Port colissimo compris. TEL: 47.64.53.51 (19 h) ou COUSTARD Eric, 4 avenue du général De Gaulle,
37000 TOURS.

Vds Calculatrice HP42S 7,2 KO SOLYER, MATRICE, STAT/PROB, etc... + 270 F CT. PRGM: logique RPN. Excel état.
Emulation HP 41 C.+2 Manuels: 500F TEL: 94.35.39.34 (le soir).

Vds Casio PB 2000 64KO RAM + ROM Basic + MD 100 (LECT 3,5"+ port RS232) + câbles +table traçante 4 couleurs FP100 Prix:2000 Frs.
Laurent (93) 83.60.27

Vds PSION series 3 128K acheté le 24/09/93 Atari TT 8MO RAM DD80 MO; Emulation spectre GCR NBS prog SUPER ETAT. TEL: 34.90.08.09 12H-13H30 18H-21H David.

Vds SHARP PC-E500, Etat neuf avec notice + emballage d'origine: 1000 Frs. Franck au 45.74.12.56 Paris 75017.

Echange ou vends cardroms, cardrams, progs, jeux (etc...) pour Portfolio. Achète carte - ROM jeux 1,2,3.
TEL: 88.95.64 14 APRES: 18H

Vds KIT transfert de données HP48 DE/A PC comprenant câble RS232 + disquette avec progs et divers logiciels jeux math LUCIEN (lyon) TEL : 78.22.23.21 (180Frs).

Cherche progm de jeux pour casio PB1000 40K sur papier. Tous frais payés.
P.HAJAS 42 rue des abricotiers,
66330 CABESTANY TEL: 68.50.66.67

Echange nombreux utilitaires et jeux pour casio graphique contre utilitaires et jeux pour HP48G. Thiébaut BERTRANT 33 rue de BITCHE 57370 PHALSBOURG.

Achète tous programmes sur DK FORMAT PC pour HP48GX écrire à :
Julien PERE LES CEDRES PORTE B 20000 AJACCIO (corse) Réponse assurée.

Vds PSION series3 256K Tableur housse cuir 1300 Frs. KINT Gérard 14000 CAEN TEL: 31.39.33.09 BUR.
Recherche listings - jeux - utils pour casio FX702P. Ecrire a: FIEY Thierry. 11/65 bis rue jude Blanckaert 59140 Dunkerque. TEL: 28.60.68.65

Vds FA80 PR FX8500 et 8000. Vds FX8500 et magnéto cassette et Vds livre HP48 en preparation. TEL: 74.68.46.42 demander Stephane HENRY.

Recherche tous programmes sur disquette PC pour HP48GX et plus particulierement un viewer et des progs de maths. Legry L 19 rue des mesanges 60600 Breuil le vert.

Vds PSION LZ64 avec adaptateur PC; 1000 Frs. Vds HP48S: 500 Frs. Vds PSION series 3 256K + tableur: 1500 Frs. Cherche programme pour HP95LX. A. COSTE TEL: 46.60.33.29

Vds FOLIOLINK complet (transfert ST-PF) 300 Frs (+30 Frs de port)
Telephoner le soir au: 34.24.87.02

Achète logiciels et extension mémoire pour PSION Series 3A d' occasion : M. LABBE Laurent TEL: 94.23.08.15 Après 20 heures.

Vends livre pour HP48: Voyage au centre de la 48 version 3. 200 F. TORRES Stéphane 18, impasse du plat 11500 Quillan.

Achète pour HP48GX tous jeux utilitaires. Enfin tous ce concernant la HP48GX... TEL: 85.40.50.47 . Après 17 heures / M. GRIMLER Olivier 2 rue des sapins 68560 Hursingue.

Vds: ATARI PORFOLIO 512KO + Packalim (accus surpuissants+mini chargeur interne+ transfo charge lente ou rapide.) +5 cartes mémoire RAM 128K+étui de protection+boitier d'interface parallèle et le câble connection PC Macintosh+P.Basic 4.91 avec utilitaires DOS supplémentaire + ROM interne type 1.130 + toutes documentations. Prix: 3500 Frs. TEL: 21.76.82.00 . FAX: 21.76.82.00 .

Vds TI 59 prodrammable + doc + chargeur + module maths le tout 200Frs. TEL: 98.07.99.60 (Si absence répondeur.)

Vds organiseur casio SF-R20 256KO tableur 3D, garantie 6 mois: 2000 Frs; SHARP PC 1600 + table traçante A4 4 couleurs + lecteur disque + 32KO RAM: 1500 Frs à débattre.
TEL: 20.47.31.89 .

Vds pour HP48 carte HP solve équation library avec tetris 700 Frs carte math calcul version 3: 500 Frs. M. DECOR 345 av J.RAYNAUD 83140 six - fours. TEL: 94.34.36.37

Vds SHARP PC1360 EXT64K + carte 8K + lecteur disquette SHARP CE140. 128K + interface cassette +programme math gestion ¢ TEL:46.55.52.10-45.80.92.06 DOM.BASSES.

Vds SHARP IQ 9000 version anglaise acheté en 02/93 3000Frs à débattre. TEL: 43.34.58.42 (AP.20h). P.ENJALBAL 31 rue adam ledoux 92400 courbevoie.

Vds SHARP 3100 2MB sousgarantie cause double emploi+lecteur disque1,44MB+adaptateur série paral.+pack alim+housse: 6300 Frs. TEL: 43.87.13.35 (14h à 19h).
GIRARD LE MANS.

Achète manuel ou photocopie du manuel de SHARP PC 1360. Contacter Régis
TEL : 78.95.35.17 Lyon.

Vds carte PACMATH pour HP48SX: 650 Frs (achetée 890 Frs) cause achat HP48GX.
TEL: 56.75.96.81 Gilles GONZALEZ, 3 rue Ed. Branly, 33140 Villenauve d'Ornon.

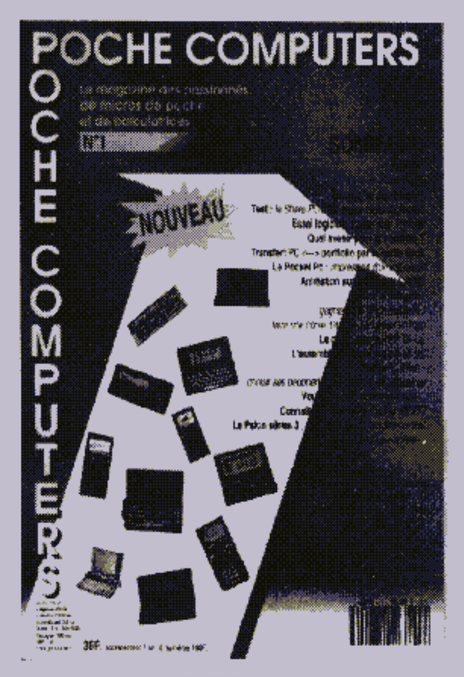
Vds HP28S excellent état + 2 livres : "la maîtrise de la HP28S". Le tout 700F plus une TI81 200F. Tel 43.27.42.53 demander LILIANE de 13h à 15h et après 20h.

Possesseur HP95LX cherche contact avec passionné expérimenté ou club pour conseils et échanges Tel 78.62.82.62 LYON après 20h.

Le .. / .. / 1994 **Bon à découper, à recopier ou à photocopier.**

Placez ici votre texte d'annonce, 1 caractère ou espace par case. N'oubliez pas vos coordonnées !

Demandez nos anciens numéros : vous pouvez les obtenir directement chez vous pour 35F + 4F de port par numéro (35F+10F de port hors de FRANCE)



N°1

N°2

GAGNEZ UN ABONNEMENT !

Vous avez conçu un programme ou une application originale ? Vous avez des idées intéressantes ? Expédiez-nous votre listing ou dessin. Si possible tapé à la machine ou sorti sur imprimante, ou encore mieux, sur disquette PC compatible (peu importe le format : Ascii -Word - Works etc... ceci nous évite de longues frappes à la main toujours fastidieuses), on le publiera certainement, et vous contribuerez à la qualité de ce magazine, pour le plaisir de tous !

Aidez-nous ! Abonnez-vous !

BULLETIN D'ABONNEMENT à POCHE COMPUTERS

(à recopier, ou à photocopier, ou à découper et à expédier à)
POCHE COMPUTERS - 42220 - BURDIGNES - FRANCE

Mme/M ..... Prénom .....
Adresse ..... Code postal ..... Ville .....
Profession (facultatif) ..... Tél ..... Fax .....
Je possède ou je désirerais posséder un ordinateur ou une calculatrice de marque .....
Je suis intéressé par tels ou tels produits .....

☐ Oui, je désire m'abonner pour 6 numéros à POCHE COMPUTERS à partir du numéro ☐ inclus et je règle la somme de 180F (215F hors de FRANCE). J'ai bien noté que je recevrai chaque numéro chez moi en franco de port, quelques jours avant sa parution en kiosque.

☐ Je ne désire qu'un ou plusieurs numéros : No ☐ ☐ ☐ et je règle la somme de 35F + 4F de port par numéro demandé. (35F+10F de port hors de France)

Règlement par ☐ chèque ☐ mandat PTT

chèques non acceptés hors de FRANCE
paiement par mandat ou carte

Indiquez ici vos impressions, soyez assurés que l'on vous lira avec attention.

.....
.....
.....
.....

POCHE COMPUTERS

BON DE COMMANDE

Tarif TTC au 1juin 1994

ce tarif remplace et annule les précédents

DESIGNATION	Prix U	Nb	Total
SHARP			
PC 3000 1MB	3450F		
PC 3000 2MB	3990F		
PC 3000 4MB	5990F		
Pack 1 : fait passer le PC 3000 1MB à 2MB	1290F		
Pack 2 : PC 3000 2MB à 4MB (expé. le SHARP)	2790F		
Lecteur de disquettes 1,44MB	2495F		
Alimentation secteur 1 ampère	165F		
Câble parallèle (imprimante)	295F		
Câble laplink (transfert PC et MAC)	295F		
Accus 1200mA (unitaire)	46F		
par 3	129F		
par 6 pour lecteur de disquettes	240F		
Etui de protection toile matelassée	295F		
Packalimentation (alim secteur +			
chargeur interne + 3 accus 1200mA)	380F		
Cartes mémoire : voir page suivante			
Organiseur IQ 9200 F	5590F		
PCE500 carte mémoire sauvegardée 128K	495F		
PCE500 carte mémoire sauvegardée 256K	690F		
HEWLETT PACKARD			
HP 48GX	2390F		
Câble de liaison HP <--> PC (avec soft)	230F		
Câble liaison HP <--> Minitel (avec soft) juillet	245F		
Cartes mémoire ram pour 48SX-GX			
48HP128 (1x128K)	495F		
48HP256 (2x128K)	795F		
48HP512 (4x128K)	1295F		
48HP1024 (8x128K)	2195F		
Cartes mémoire rom pour 48SX-GX			
UTILS 1 ou JEUX 1 ou HPMANIAC 1	420F		
par 2 cartes (entourez celles choisies)	790F		
par 3 cartes	1100F		
Carte PACMATH	890F		
Lecteur de disquettes pour HP48/ HP95 1,44MB	2650F		
Convertisseur A/D pour HP48 - 2 fois 16bits	TEL		
Protecteur rigide environnement hostile HP48	TEL		
HP 95LX 1MB	3990F		
TOPOLAND (soft carnet de terrain)	TEL		
HP 100LX 1MB	4790F		
HP 100LX 2MB	6390F		
Câble de connection HP <--> PC	230F		
Câble de connection HP <--> Mac	230F		
Kit de connection HP95/100 <--> PC	760F		
Kit de connection HP95/100 <--> PC	760F		
Kit de connection HP95/100 <--> Mac	760F		
convertisseur série/parallèle	660F		
Pack 3 : fait passer votre HP100LX 1MB à 2MB	1800F		
nécessite l'envoi de l'appareil (dispo juillet)			
Carte Fax-modem PCMCIA pour HP95 (4800b)	1760F		
Carte Fax-modem PCMCIA pour HP100 (9600b)	2490F		
Cartes mémoire PCMCIA (voir page suivante)			

	Prix U	Nb	Total
PSION SERIES 3/3a			
PSION series 3a 256K (version anglaise)	3190F		
PSION series 3a 512K (version anglaise)	3490F		
Lecteur de disquettes 1,44MB	2790F		
SSD RAM 128K	735F		
SSD RAM 256K	890F		
SSD RAM 512K	1490F		
SSD RAM 1MB	2920F		
SSD FLASH 128K	520F		
SSD FLASH 256K	720F		
SSD FLASH 512K	1220F		
SSD FLASH 1MB	2090F		
SSD FLASH 2MB	2890F		
Liaison série pour PC ou Mac	790F		
Liaison parallèle	440F		
Pack 4 : fait passer le series 3 128K à 256K	300F		
Pack 5 : fait passer le series 3a 256K à 512K	390F		
Pack 6 : fait passer le series 3a 256K à 1MB	TEL		
Pack 7 : fait passer le series 3a 512K à 1MB	TEL		
Tous ces packs nécessitent l'envoi du Series 3			
Kit développeur standard pour series 3/3a	2390F		
Kit développeur professionnel pour series 3/3a	3590F		
FAX-MODEM (dispo mi-juin) 9600bds	2850F		
Emulateur minitel (soft + câble)	1290F		
Emulateur minitel (soft + câble) + modem V23	1890F		
Option émulateur minitel sur SSD 128K : supplément	520F		
PERIPHERIQUES SOFT (en anglais)			
Winlink Windows (complément de serial link)	430F		
Nouveau soft pour Mac link (disquette)	195F		
VT emulator (sur disquette)	650F		
SOFTS (en anglais + mise en route français)			
Psion Money pour Series 3a (sur SSD)	595F		
Finance Pack series 3	550F		
Finance pack series 3a	550F		
JEUX 1	550F		
JEUX 2	550F		
JEUX 3	550F		
3 flight simulator Planner (sur SSD 512K)	1590F		
Professional Finance	790F		
Paint and compose Series 3/3a	720F		
NOUVEAUX POCKETS HANBIT			
Palmtopia 286 - HBC 100 1MB ram	4990F		
Palmtopia 286 - HBC 120 2MB ram	5390F		
Palmtopia 286 - HBC 100s 1MB ram + sound blaster	5590F		
Palmtopia 286 - HBC 120s 2MB ram + sound blaster	5990F		
Palmnote 386 - HBC 1320 2MB ram	8850F		
Palmnote 386 - HBC 1340 4MB ram	9990F		
Palmnote 486 - HBC 1420 2MB ram	10900F		
Palmnote 486 - HBC 1440 4MB ram	12400F		
Palmbook 486 - HBC 4413 4MB ram - disque 130MB	15990F		
Palmbook 486 - HBC 4413 4MB ram - disque 260MB	17550F		
Système portatif intégrant codes barres et imprimante	19500F		
IC-MAN HRC-200	1590F		
IC-MAN HRC-300	1790F		

sous total A ---------->		sous total B ---------->	

MICROCARDS - 42220 - BURDIGNES - Tel 77.39.68.13 Fax 77.39.19.60

POCHE COMPUTERS

BON DE COMMANDE

Tarif TTC au 1er juin 1994
ce tarif remplace et annule les précédents

DESIGNATION	Prix U	Nb	Total		DESIGNATION	Prix U	Nb	Total
ATARI PORTFOLIO					2MB	2690F		
Portfolio neuf 128K	1750F				4MB doublée stacker (spéciale pour HP100LX)	3490F		
Portfolio neuf 256K	1990F				Logiciel STACKER 4.0 (pour doubler la taille des cartes)	1190F		
Portfolio neuf 512K	2510F				cartes FLASH EPROM			
Portfolio occasion révisé atelier garanti 3 mois	900F				256KB	620F		
Cardram 128K (carte de crédit)	495F				512KB	920F		
Cardram 256K (carte de crédit)	690F				1MB	1490F		
Cardram 512K (carte de crédit)	960F				2MB	2490F		
Cardram 1024K	1490F				5MB (HP 100LX et Omnibook)	3990F		
EMI 256K	520F				10MB (HP 100LX et Omnibook)	5990F		
EMI 384K	790F				20MB (HP 100LX et Omnibook)	8990F		
EMI 512K	970F				disques durs portatifs externes : (14cmx7cmx2cm ~)			
Packalimentation	380F				s'adaptent sur la plupart des pockets			
Alimentation de sauvegarde	245F				mini disque dur portatif sur port // 170MB (15ms)	3750F		
Câble de liaison imprimante parallèle	80F				mini disque dur portatif sur port // 260MB (15ms)	4750F		
Interface mixte parallèle et série	590F				cartes PREPROGRAMMEES (ROM-RAM en anglais)			
Câble de liaison PC	80F				compatibles HP-SHARP-OLIVETTI-AMSTRAD etc...			
Lecteur de cartes mémoire sur PC	1290F				mise en route en français avec traduction simplifiée			
Manuel technique et de programmation (200pages)	250F				Astrotutor (astronomie)	1190F		
Etui de protection toile matelassée	295F				Bearings (pour pilotes-skippers-radio amateurs)	1190F		
Changement de ROM nouvelle version 1.130	370F				Finance card (calcul d'intérêt-amortissements-etc)	1190F		
SOFTS (sur cartes rom)					Stock organiser (gestion complète de stocks + mém)	1590F		
Turbo Forth (version complète PC + PORTFOLIO)	795F				LIVRES			
PBASIC 4.91	470F				Pour PSION Organiser II (en anglais)			
PFSOLVER	380F				Psion Organiser II Deciphered	290F		
PBASE	490F				File handing and other programs for Organiser II	290F		
Gestion familiale	490F				Machine code programming on the Organiser II	290F		
Le contrôleur	490F				Pour PSION Series 3/3a (en anglais)			
Jeux1	380F				First steps in programming the Series 3	290F		
Jeux 2	380F				Serious programming the Series 3	290F		
Utils 1	380F				Serious programming the Series 3a	290F		
Utils 2	380F				Introduction to using Series 3	290F		
Utils 3	380F				Graphics programming the Series 3	290F		
Macfolio	380F				Pour HP 48SX/GX			
Transfert Atari st <--> Portfolio	380F				Les secrets de la HP48G/GX	220F		
REPARATIONS de PORTFOLIO					Externals et assembleur	220F		
forfait changement liaison écran-clavier	320F				Mathez la HP48G/GX	200F		
forfait changement d'alimentation	160F				La maîtrise de la HP 48S/SX	200F		
forfait changement d'écran (occasion)	450F				Les secrets de la HP 48S/SX	220F		
forfait révision générale (écran-clavier-carte mère)	150F				Pour SHARP			
CARTES MEMOIRE PCMCIA (pour HP95/100-Sharp 3000					manuel langage machine pour PC1350/1403 (anglais)	390F		
OLIVETTI QUADERNO/PHILOS-AMSTRAD PENPAD-etc)					manuel langage machine PC1500 (anglais)	250F		
cartes SRAM					DIVERS			
256KB	690F				Mémo formulaire	90F		
512KB	950F				Mémotech électronique	260F		
1MB	1590F				Mémotech électrotechnique	260F		

sous total C ---------->		sous total D ---------->	

Port colissimo + assurance :
pour une valeur jusqu'à 300F ttc ----------> 25F
pour une valeur jusqu'à 1500F ttc ----------> 45F
pour une valeur jusqu'à 4000F ttc ----------> 60F
au delà, 45F/60F par tranche de 1500F/4000F
Supplément pour chronopost : ----------> + 120F
Supplément pour contre remboursement ---------> 45F

Mme/M : NOM .....
Adresse .....
.....
TEL ..... FAX .....
Je possède un ordinateur de marque ..... Type .....
.....

Total général (sous total A+sous total B +sous total C + sous total D + port)

Je règle par : mandat PTT ☐ chèque ☐ contre remboursement ☐
Date le . . / . . / 1994 signature

Impressions Dumas, 42100 Saint-Étienne – N° d'imprimeur : 31929

POCHE COMPUTERS

ATARI PORTFOLIO

MICROCARDS - 42220 - BURDIGNES - FRANCE
tél (33) 77.39.68.13 - fax (33) 77.39.19.60

Tarif TTC au 1/06/1994 (ce tarif remplace et annule les précédents)

HARDWARE

Cartes mémoire RAM (format carte de crédit)

Cardram CCR 128 495,00 F
Cardram CCR 256 690,00 F
Cardram CCR 512 960,00 F

Cartes mémoire ROM (format carte de crédit)

Cardrom CCR 128 430,00 F
Cardrom CCR 256 580,00 F
Cardrom CCR 512 810,00 F

(nota: les cardroms sont programmables uniquement avec un programmateur spécial)

Portfolios neufs

Portfolio 128K 1750,00 F
Portfolio 256K 1990,00 F
Portfolio 384K 2260,00 F
Portfolio 512K 2510,00 F

Portfolio occasion (garantie de 3 mois- en parfait état de marche, entièrement révisé atelier) 900F

Extensions mémoire interne (augmentent la mémoire interne du Portfolio ; nécessite l'envoi de votre appareil)

EMI 256K (ajoute 128K au Portfolio) 520,00 F
EMI 384K (ajoute 256K au Portfolio) 790,00 F
EMI 512K (ajoute 384K au Portfolio) 980,00 F

Packalimentation Système remplaçant les piles par des accus très puissants (850 ou 1200 mA) et se rechargeant dans le Portfolio par un bloc secteur. Ce système comprend la fourniture des 3 éléments : chargeur interne (nécessite donc l'envoi du Portfolio) + accus + alimentation : Très fiable. Amortissement du système en quelques mois. Nécessite l'envoi du Portfolio.

Le packalim avec accus de 850 mA 320,00 F
Le packalim avec accus de 1200 mA 420,00 F

Alimentation de sauvegarde Complémentaire du Packalim, cette alim permet de sauvegarder les données du Portfolio en cas de problème d'alimentation générale (plantage, micro coupures etc...) Un micro accumulateur vient prendre immédiatement le relais des piles ou accus (conservation des données durant plusieurs mois sans aucune source d'alimentation ; nécessite l'envoi du Portfolio).

Alimentation de sauvegarde 245,00 F

Interface mixte parallèle/série 590,00F liaison facile entre un Portfolio et un PC ou une imprimante. Permet la communication également avec d'autres périphériques (MAC avec Macfolio)

Câble de liaison PC 80,00 F

Câble de liaison imprimante 80,00 F

Changement de Rom interne Remplacement des versions 1.052 ou 1.072 du bios par une version récente 1.130 ; permet d'éviter les plantages et autres erreurs division par zéro ! BIOS 370,00 F

Accumulateurs surpuissants ! Ces nouveaux accumulateurs possèdent une puissance de 1200 mA avec une tension maxi de 1,55 volt, et pour une taille standard de type R6. Remplacent les piles du Portfolio et en augmentent son autonomie de 40%. Ils sont utilisables également dans d'autres appareils, ainsi qu'en photo et en modélisme. Le 850mA prix u. : 23 F - le 1200mA prix u. : 46 F

Lecteur de cartes mémoire sur PC compatible : 1290F

SERVICE APRES-VENTE ET REPARATION DES ORDINATEURS DE POCHE

Devis gratuit sur simple demande

Tarif forfaitaire pour le Portfolio
: changement liaison écran-clavier : 320F
: changement d'écran (occasion) : 450F
: changement d'alimentation : 160F

SOFTWARE

PBASIC Dernière version 4.91 de BJ.GLEASON avec de nombreux utilitaires Dos. Compatible GWBASIC, ce langage est particulièrement adapté au Portfolio. Il est fourni avec une documentation pour débutants avec des exemples simples. N'occupant que 40K, il fonctionne sans problème avec un Portfolio standard.
prix : 470,00 F (sur carte rom)

ASTROPLAN Superbe logiciel d'Astrologie pour amateur ou professionnel. Fonctionne sur Portfolio standard. prix: 990,00 F (sur carte rom)

PFSOLVER de R.NOMI. BEGUIN
Avec ce module,vous n'aurez plus rien à envier à HP !
Graphisme et tracés de courbes ; fonctions exponentielles et polaires etc... Ce soft de qualité a été élaboré par un élève-ingénieur. Fournis sur carte Rom et fonctionnent sur Portfolio standard. Documentation explicite. Prix : 380,00 F

PBASE
Superbe base de données pour Portfolio. Gére jusqu'à 30 champs différents. Indispensable ! Fonctionne sur Portfolio standard et fournie sur carte Rom. Documentation détaillée et d'apprentissage.
Prix : 490,00 F

GESTION FAMILIALE
Pour gérer votre budget - cartes de crédit - chèquier avec établissement des comptes et sortie sur imprimante. Fonctionne sur Portfolio standard. Fournie sur carte Rom. Prix : 490,00 F

LE CONTROLEUR
Puissant logiciel de gestion d'emploi du temps ; avec visites clients, horaires, relances, factures etc... Sur Portfolio standard et fournie sur carte Rom. Prix : 590,00 F

JEUX 1 et 2 2 cartes mémoire ROM ludiques exploitant pleinement les possibilités du Portfolio. Jeux de réflexion, d'adresse etc... Fonctionnent sur Portfolio standard. Fournies sur carte Rom.
Prix JEUX 1 : 380,00 F
Prix JEUX 2 : 380,00 F

UTILS 1,2,3 3 trois cartes ROM bourrées d'utilitaires du dos pour Portfolio, ainsi que de programmes divers. Elles fonctionnent toutes sur Portfolio standard.
Prix UTILS 1 : 380,00 F
Prix UTILS 2 : 380,00 F
Prix UTILS 3 : 380,00 F

MACFOLIO de R.THIBERT Avec cet ensemble, soft + câble, la communication entre le Portfolio et un Macintosh devient possible. Fonctionne avec tous les modèles de Mac. Prix : 380,00 F

Transfert ST/TT <-->PF Ensemble câble + soft permettant la liaison entre le Portfolio et un Atari ST/TT. Prix : 380,00 F

ETUI de protection Magnifique étui en toile matelassée spécialement étudié pour le Portfolio: 295,00F

Garantie totale pendant une année sur tous nos produits.

Frais de port colissimo+assurance =
25F (PTT colieco) jusqu'à 300F ttc de matériel -
45F jusqu'à 1500F ttc de matériel -
60F jusqu'à 4000F ttc de matériel et par tranche au delà.
45,00F de supplément de contre remboursement
et supplément de 120,00F pour un chronopost.

CATALOGUE SUR SIMPLE DEMANDE

VENTE PAR CORRESPONDANCE EXCLUSIVEMENT

(ouverture du lundi au vendredi inclus : de 9h à 13h et de 15h à 19h)

Copyright 1994 MICROCARDS

Document non contractuel : les prix peuvent être modifiés sans préavis

Vous recherchez un spécialiste en pockets ?

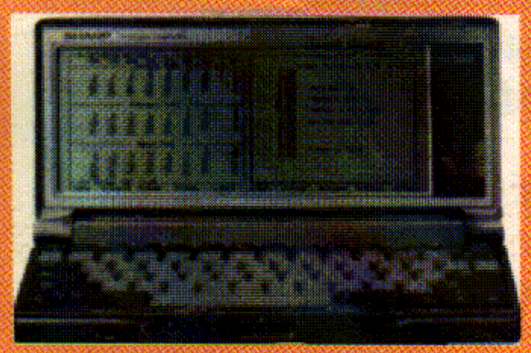
Vous avez de la chance... c'est notre domaine !

Le HP 100LX
1MB et 2MB de mémoire
écran de 80 col par 25-CGA
MS DOS 5.0 - Lotus 123
1 slot PCMCIA type II
dim : 16cm x 8,6 x 2,5
poids 320g

Le PALMBOOK 486
processeur 486SLC/25
4MB de mémoire --->16MB
écran de 80 col par 25 VGA
rétroéclairé 7pouces- 64NG
sortie moniteur écran couleur
track ball - disque dur 260MB
MS DOS 6.2 - Windows 3.1
1 slot PCMCIA type III ou
2 de type II -reconnaissance vocale
intégrée dim : 24cm x 13cm x 3,9cm-
poids 990g

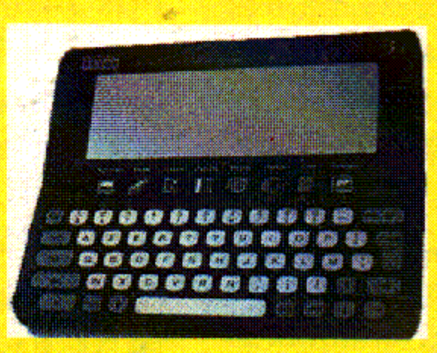
NOUVEAU pocket disponible en version processeur 286 - 386 ou 486 ou version codes à barres et mini imprimante intégrés

Cartes mémoires SRAM ou FLASH type PCMCIA de différentes tailles ...



Le SHARP PC 3000
1MB ou 2MB de mémoire
écran de 80 col par 25 CGA
MS DOS 3.3 - 7 logiciels intégrés
2 slots PCMCIA type I
dim : 22cm x 11 x 2,5
poids 480g

ET AUSSI TOUS LES ACCESSOIRES LIES A CES APPAREILS



Le PSION series 3a 256K ou 512K de mémoire
écran de 79 col par 17-CGA - DOS constructeur
logiciels intégrés - 2 slots cartes mémoire
dimensions : 16,5cm x 8,5 x 2,2 - poids 270g

Les nouvelles cartes préprogrammées PCMCIA type I, fonctionnant sur la plupart des pockets (HP95-100LX - SHARP PC 3000 - POCKET PC - OMNIBOOK etc... et tous les portables : TOSHIBA - COMPAQ - OLIVETTI etc...)
Ces cartes ROM ou mixtes (ROM-RAM) abordent des domaines aussi variés que : l'assurance - la physique - le médical - la construction - la finance - la restauration - l'automobile - la production - le droit - les corps de métiers, et bien d'autres encore...

Mini disque dur pour tous pockets.
Se branchent sur l'interface parallèle, sans aucune carte, ni même modification du config.sys.
Ultra rapides
capacité : 170MB (15ms)
ou 260MB (15ms)
taille (14cmx7cmx2cm)

MICROCARDS - 42220 - BURDIGNES - FRANCE
tél (33) 77.39.68.13 (lignes groupées) fax (33) 77.39.19.60

voir tarif et bon de commande en pages intérieures
catalogue complet pockets et calculatrices disponible début juillet 1994 contre 15F en timbres

document non contractuel : les prix peuvent être sujets à révision sans préavis - les marques citées sont des marques déposées